User's Manual

16

# RL78/G1A

## User's Manual: Hardware

16-Bit Single-Chip Microcontrollers



Renesas Electronics
www.renesas.com

Rev.2.00 Jul 2013

# Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.

   “Standard”: Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots etc.

   “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.

   Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(2012.4)

NOTES FOR CMOS DEVICES

(1) VOLTAGE APPLICATION WAVEFORM AT INPUT PIN: Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between VIL (MAX) and VIH (MIN) due to noise, etc., the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between VIL (MAX) and VIH (MIN).

(2) HANDLING OF UNUSED INPUT PINS: Unconnected CMOS device inputs can be cause of malfunction. If an input pin is unconnected, it is possible that an internal input level may be generated due to noise, etc., causing malfunction. CMOS devices behave differently than Bipolar or NMOS devices. Input levels of CMOS devices must be fixed high or low by using pull-up or pull-down circuitry. Each unused pin should be connected to VDD or GND via a resistor if there is a possibility that it will be an output pin. All handling related to unused pins must be judged separately for each device and according to related specifications governing the device.

(3) PRECAUTION AGAINST ESD: A strong electric field, when exposed to a MOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop generation of static electricity as much as possible, and quickly dissipate it when it has occurred. Environmental control must be adequate. When it is dry, a humidifier should be used. It is recommended to avoid using insulators that easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors should be grounded. The operator should be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions need to be taken for PW boards with mounted semiconductor devices.

(4) STATUS BEFORE INITIALIZATION: Power-on does not necessarily define the initial status of a MOS device. Immediately after the power source is turned ON, devices with reset functions have not yet been initialized. Hence, power-on does not guarantee output pin levels, I/O settings or contents of registers. A device is not initialized until the reset signal is received. A reset operation must be executed immediately after power-on for devices with reset functions.

(5) POWER ON/OFF SEQUENCE: In the case of a device that uses different power supplies for the internal operation and external interface, as a rule, switch on the external power supply after switching on the internal power supply. When switching the power supply off, as a rule, switch off the external power supply and then the internal power supply. Use of the reverse power on/off sequences may result in the application of an overvoltage to the internal elements of the device, causing malfunction and degradation of internal elements due to the passage of an abnormal current. The correct power on/off sequence must be judged separately for each device and according to related specifications governing the device.

(6) INPUT OF SIGNAL DURING POWER OFF STATE : Do not input signals or an I/O pull-up power supply while the device is not powered. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Input of signals during the power off state must be judged separately for each device and according to related specifications governing the device.

# How to Use This Manual

**Readers**

This manual is intended for user engineers who wish to understand the functions of the RL78/G1A and design and develop application systems and programs for these devices.
The target products are as follows.

- 25-pin: R5F10E8x (x = A, C, D, E)
- 32-pin: R5F10EBx (x = A, C, D, E)
- 48-pin: R5F10EGx (x = A, C, D, E)
- 64-pin: R5F10ELx (x = C, D, E)

**Purpose**

This manual is intended to give users an understanding of the functions described in the **Organization** below.

**Organization**

The RL78/G1A manual is separated into two parts: this manual and the instructions edition (common to the RL78 Microcontroller).

| **RL78/G1A User's Manual (This Manual)** | **RL78 Microcontroller User's Manual Instructions** |
|---|---|
| • Pin functions<br>• Internal block functions<br>• Interrupts<br>• Other on-chip peripheral functions<br>• Electrical specifications | • CPU functions<br>• Instruction set<br>• Explanation of each instruction |

**How to Read This Manual**

It is assumed that the readers of this manual have general knowledge of electrical engineering, logic circuits, and microcontrollers.

- To gain a general understanding of functions:
  → Read this manual in the order of the **CONTENTS**. The mark "<R>" shows major revised points. The revised points can be easily searched by copying an "<R>" in the PDF file and specifying it in the "Find what:" field.
- How to interpret the register format:
  → For a bit number enclosed in angle brackets, the bit name is defined as a reserved word in the assembler, and is defined as an sfr variable using the #pragma sfr directive in the compiler.
- To know details of the RL78G1A Microcontroller instructions:
  → Refer to the separate document **RL78 Microcontroller Instructions User's Manual (R01US0015E)**.

**Conventions**

| | | |
|---|---|---|
| Data significance: | Higher digits on the left and lower digits on the right | |
| Active low representations: | $\overline{\times\times\times}$ (overscore over pin and signal name) | |
| **Note**: | Footnote for item marked with **Note** in the text | |
| **Caution**: | Information requiring particular attention | |
| **Remark**: | Supplementary information | |
| Numerical representations: | Binary | ···×××× or ××××B |
| | Decimal | ···×××× |
| | Hexadecimal | ···××××H |

**Related Documents** The related documents indicated in this publication may include preliminary versions. However, preliminary versions are not marked as such.

**Documents Related to Devices**

| Document Name | Document No. |
|---|---|
| RL78/G1A User's Manual Hardware | This manual |
| RL78 family User's Manual: Software | R01US0015E |

**Documents Related to Flash Memory Programming**

| Document Name | Document No. |
|---|---|
| PG-FP5 Flash Memory Programmer User's Manual | R20UT0008E |

**Caution The related documents listed above are subject to change without notice. Be sure to use the latest version of each document when designing.**

**Other Documents**

| Document Name | Document No. |
|---|---|
| Renesas MPUs & MCUs RL78 Family | R01CP0003E |
| Semiconductor Package Mount Manual | **Note** |
| Quality Grades on NEC Semiconductor Devices | C11531E |
| Guide to Prevent Damage for Semiconductor Devices by Electrostatic Discharge (ESD) | C11892E |
| Semiconductor Reliability Handbook | R51ZZ0001E |

**Note** See the “Semiconductor Package Mount Manual” website (http://www.renesas.com/products/package/manual/index.jsp).

**Caution The related documents listed above are subject to change without notice. Be sure to use the latest version of each document when designing.**



Caution: This product uses SuperFlash® technology licensed from Silicon Storage Technology, Inc.

**CONTENTS**

# CHAPTER 1 OUTLINE

## <R> 1.1 Features

Ultra-low power consumption technology

- $V_{DD}$ = single power supply voltage of 1.6 to 3.6 V which can operate a 1.8 V device at a low voltage
- HALT mode
- STOP mode
- SNOOZE mode

RL78 CPU core

- CISC architecture with 3-stage pipeline
- Minimum instruction execution time: Can be changed from high speed (0.03125 µs: @ 32 MHz operation with high-speed on-chip oscillator) to ultra-low speed (30.5 µs: @ 32.768 kHz operation with subsystem clock)
- Address space: 1 MB
- General-purpose registers: (8-bit register × 8) × 4 banks
- On-chip RAM: 2 to 4 KB

Code flash memory

- Code flash memory: 16 to 64 KB
- Block size: 1 KB
- Prohibition of block erase and rewriting (security function)
- On-chip debug function
- Self-programming (with boot swap function/flash shield window function)

Data flash memory

- Data flash memory: 4 KB
- Back ground operation (BGO): Instructions can be executed from the program memory while rewriting the data flash memory.
- Number of rewrites: 1,000,000 times (TYP.)
- Voltage of rewrites: $V_{DD}$ = 1.8 to 3.6 V

High-speed on-chip oscillator

- Select from 32 MHz, 24 MHz, 16 MHz, 12 MHz, 8 MHz, 6 MHz, 4 MHz, 3 MHz, 2 MHz, and 1 MHz
- High accuracy ±1.0 % ($V_{DD}$ = 1.8 to 3.6 V, $T_A$ = -20 to +85°C)

Operating ambient temperature

- $T_A$ = -40 to +85°C (A: Consumer applications)
- $T_A$ = -40 to +105°C (G: Industrial applications)

Power management and reset function

- On-chip power-on-reset (POR) circuit
- On-chip voltage detector (LVD) (Select interrupt and reset from 12 levels)

DMA (Direct Memory Access) controller

- 2 channels
- Number of clocks during transfer between 8/16-bit SFR and internal RAM: 2 clocks

Multiplier and divider/multiply-accumulator

- 16 bits × 16 bits = 32 bits (Unsigned or signed)
- 32 bits ÷ 32 bits = 32 bits (Unsigned)
- 16 bits × 16 bits + 32 bits = 32 bits (Unsigned or signed)

Serial interface

- CSI: 2 to 6 channels
- UART/UART (LIN-bus supported): 2 or 3 channels
- $I^2C$/Simplified $I^2C$ communication: 2 to 7 channels

Timer

- 16-bit timer: 8 channels
- 12-bit interval timer: 1 channel
- Real-time clock: 1 channel (calendar for 99 years, alarm function, and clock correction function)
- Watchdog timer: 1 channel (operable with the dedicated low-speed on-chip oscillator)

A/D converter

- 8/12-bit resolution A/D converter ($V_{DD}$ = 1.6 to 3.6 V)
- Analog input: 13 to 28 channels
- Internal reference voltage (1.45 V) and temperature sensor[Note 1]

I/O port

- I/O port: 19 to 56 (N-ch open drain I/O [withstand voltage of 6 V]: 2 to 4,
  N-ch open drain I/O [$V_{DD}$ withstand voltage[Note 2]/$EV_{DD}$ withstand voltage[Note 3]]: 6 to 12)
- Can be set to N-ch open drain, TTL input buffer, and on-chip pull-up resistor
- Different potential interface: Can connect to a 1.8/2.5 V device
- On-chip key interrupt function
- On-chip clock output/buzzer output controller

Others

- On-chip BCD (binary-coded decimal) correction circuit

**Notes 1.** Can be selected only in HS (high-speed main) mode

**2.** Products with 25 to 48 pins

**3.** Products with 64 pins

**Remark** The functions mounted depend on the product. See **1.6 Outline of Functions**.

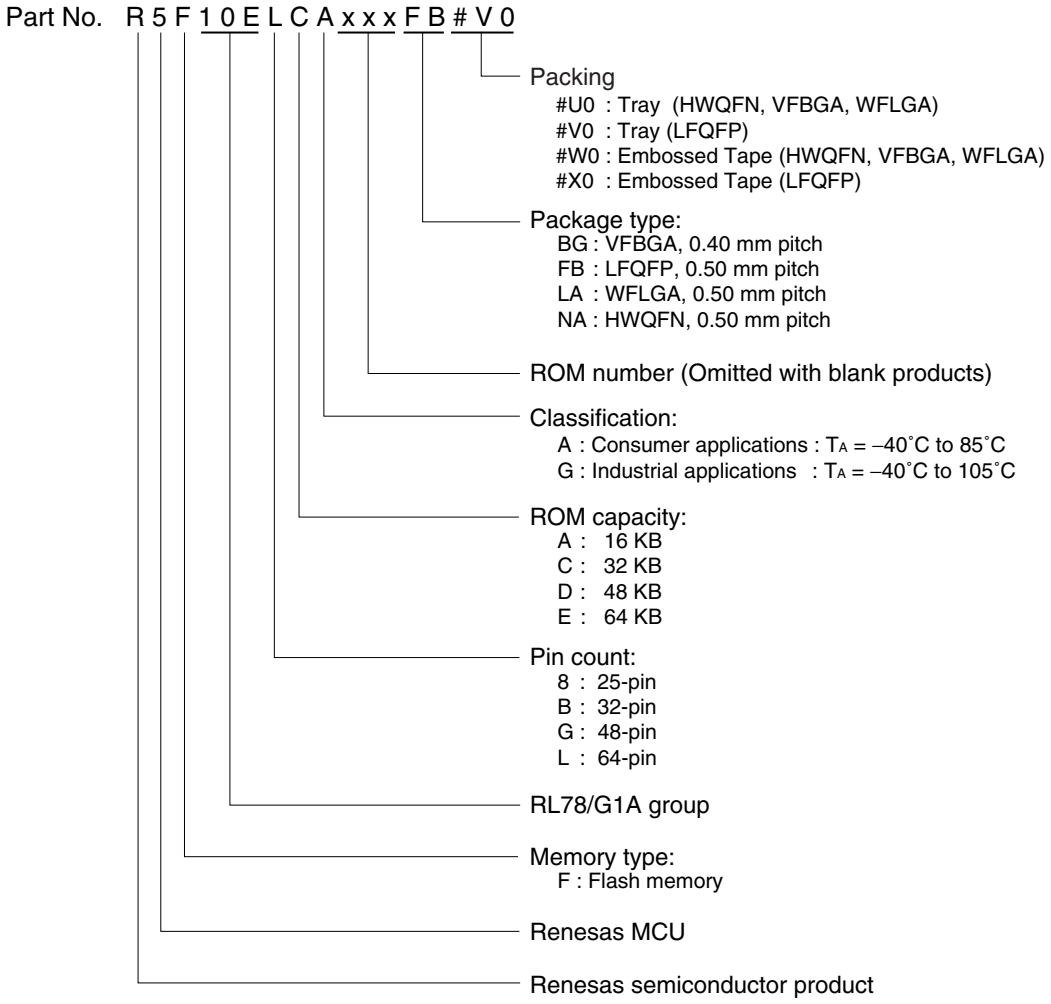
○ ROM, RAM capacities

| Flash ROM | Data flash | RAM | RL78/G1A | | | |
|---|---|---|---|---|---|---|
| | | | 25 pins | 32 pins | 48 pins | 64 pins |
| 64 KB | 4 KB | 4 KB Note | R5F10E8E | R5F10EBE | R5F10EGE | R5F10ELE |
| 48 KB | 4 KB | 3 KB | R5F10E8D | R5F10EBD | R5F10EGD | R5F10ELD |
| 32 KB | 4 KB | 2 KB | R5F10E8C | R5F10EBC | R5F10EGC | R5F10ELC |
| 16 KB | 4 KB | 2 KB | R5F10E8A | R5F10EBA | R5F10EGA | – |

**Note** This is about 3 KB when the self-programming function and data flash function are used. (For details, see **CHAPTER 3**)

## 1.2 List of Part Numbers

**Figure 1-1. Part Number, Memory Size, and Package of RL78/G1A**




**Caution The part number above is valid as of when this manual was issued. For the latest part number, see the web page of the target product on the Renesas Electronics website.**

**Table 1-1. List of Ordering Part Numbers**

| Pin count | Package | Fields of Application[Note 1] | Ordering Part Number |
|---|---|---|---|
| 25 pins | 25-pin plastic WFLGA (3 × 3 mm, 0.5 mm pitch) | A | R5F10E8AALA#U0, R5F10E8CALA#U0, R5F10E8DALA#U0, R5F10E8EALA#U0, R5F10E8AALA#W0, R5F10E8CALA#W0, R5F10E8DALA#W0, R5F10E8EALA#W0 |
| | | G[Note 2] | R5F10E8AGLA#U0, R5F10E8CGLA#U0, R5F10E8DGLA#U0, R5F10E8EGLA#U0, R5F10E8AGLA#W0, R5F10E8CGLA#W0, R5F10E8DGLA#W0, R5F10E8EGLA#W0 |
| 32 pins | 32-pin plastic HWQFN (5 × 5 mm, 0.5 mm pitch) | A | R5F10EBAANA#U0, R5F10EBCANA#U0, R5F10EBDANA#U0, R5F10EBEANA#U0, R5F10EBAANA#W0, R5F10EBCANA#W0, R5F10EBDANA#W0, R5F10EBEANA#W0 |
| | | G | R5F10EBAGNA#U0, R5F10EBCGNA#U0, R5F10EBDGNA#U0, R5F10EBEGNA#U0, R5F10EBAGNA#W0, R5F10EBCGNA#W0, R5F10EBDGNA#W0, R5F10EBEGNA#W0 |
| 48 pins | 48-pin plastic LFQFP (7 × 7 mm, 0.5 mm pitch) | A | R5F10EGAAFB#V0, R5F10EGCAFB#V0, R5F10EGDAFB#V0, R5F10EGEAFB#V0, R5F10EGAAFB#X0, R5F10EGCAFB#X0, R5F10EGDAFB#X0, R5F10EGEAFB#X0 |
| | | G | R5F10EBAGNA#V0, R5F10EBCGNA#V0, R5F10EBDGNA#V0, R5F10EBEGNA#V0, R5F10EBAGNA#X0, R5F10EBCGNA#X0, R5F10EBDGNA#X0, R5F10EBEGNA#X0 |
| | 48-pin plastic HWQFN (7 × 7 mm, 0.5 mm pitch) | A | R5F10EGAANA#U0, R5F10EGCANA#U0, R5F10EGDANA#U0, R5F10EGEANA#U0, R5F10EGAANA#W0, R5F10EGCANA#W0, R5F10EGDANA#W0, R5F10EGEANA#W0 |
| | | G | R5F10EGAGNA#U0, R5F10EGCGNA#U0, R5F10EGDGNA#U0, R5F10EGEGNA#U0, R5F10EGAGNA#W0, R5F10EGCGNA#W0, R5F10EGDGNA#W0, R5F10EGEGNA#W0 |
| 64 pins | 64-pin plastic LFQFP (10 × 10 mm, 0.5 mm pitch) | A | R5F10ELCAFB#V0, R5F10ELDAFB#V0, R5F10ELEAFB#V0, R5F10ELCAFB#X0, R5F10ELDAFB#X0, R5F10ELEAFB#X0 |
| | | G | R5F10ELCGFB#V0, R5F10ELDGFB#V0, R5F10ELEGFB#V0, R5F10ELCGFB#X0, R5F10ELDGFB#X0, R5F10ELEGFB#X0 |
| | 64-pin plastic VFBGA (4 × 4 mm, 0.4 mm pitch) | A | R5F10ELCABG#U0, R5F10ELDABG#U0, R5F10ELEABG#U0, R5F10ELCABG#W0, R5F10ELDABG#W0, R5F10ELEABG#W0 |
| | | G[Note 2] | R5F10ELCGBG#U0, R5F10ELDGBG#U0, R5F10ELEGBG#U0, R5F10ELCGBG#W0, R5F10ELDGBG#W0, R5F10ELEGBG#W0 |

**Notes 1.** For the fields of application, see **Figure 1-1 Part Number, Memory Size, and Package of RL78/G1A**.

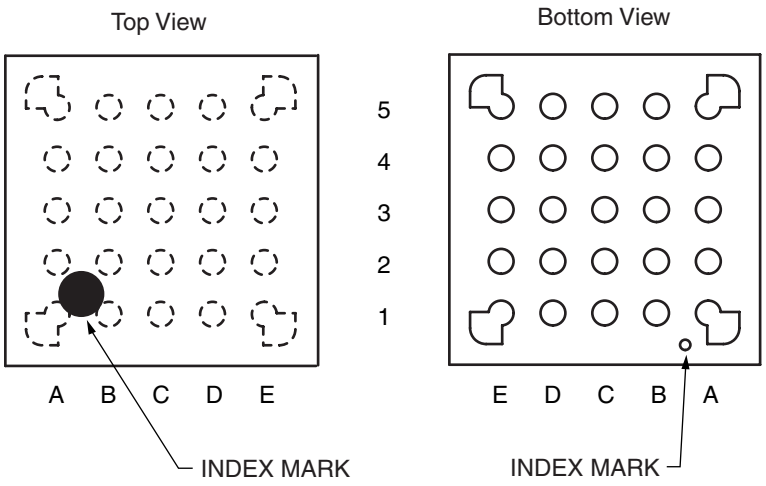
**2.** In planning

**Caution The part number above is valid as of when this manual was issued. For the latest part number, see the web page of the target product on the Renesas Electronics website.**

## 1.3 Pin Configuration (Top View)

### 1.3.1 25-pin products

- 25-pin plastic WFLGA (3 × 3 mm, 0.50 mm pitch)




| | A | B | C | D | E | |
|---|---|---|---|---|---|---|
| 5 | P40/TOOL0 | $\overline{\text{RESET}}$ | P03/ANI16/ RxD1/TO00/ (KR1) | P23/ANI3/ (KR3) | $AV_{SS}$ | 5 |
| 4 | P122/X2/ EXCLK | P137/INTP0 | P02/ANI17/ TxD1/TI00/ (KR0) | P22/ANI2/ (KR2) | $AV_{DD}$ | 4 |
| 3 | P121/X1 | $V_{DD}$ | P21/ANI1/ $AV_{REFM}$ | P11/ANI20/ SI00/SDA00/ RxD0/ TOOLRxD | P10/ANI18/ SCK00/SCL00 | 3 |
| 2 | REGC | $V_{SS}$ | P30/ANI27/ SCK11/SCL11/ INTP3 | P51/ANI25/ SO11/INTP2 | P50/ANI26/ SI11/SDA11 INTP1 | 2 |
| 1 | P60/SCLA0 | P61/SDAA0 | P31/ANI29/TI03/ TO03/PCLBUZ0 /INTP4 | P12/ANI21/ SO00/TxD0/ TOOLTxD | P20/ANI0/ $AV_{REFP}$ | 1 |
| | A | B | C | D | E | |

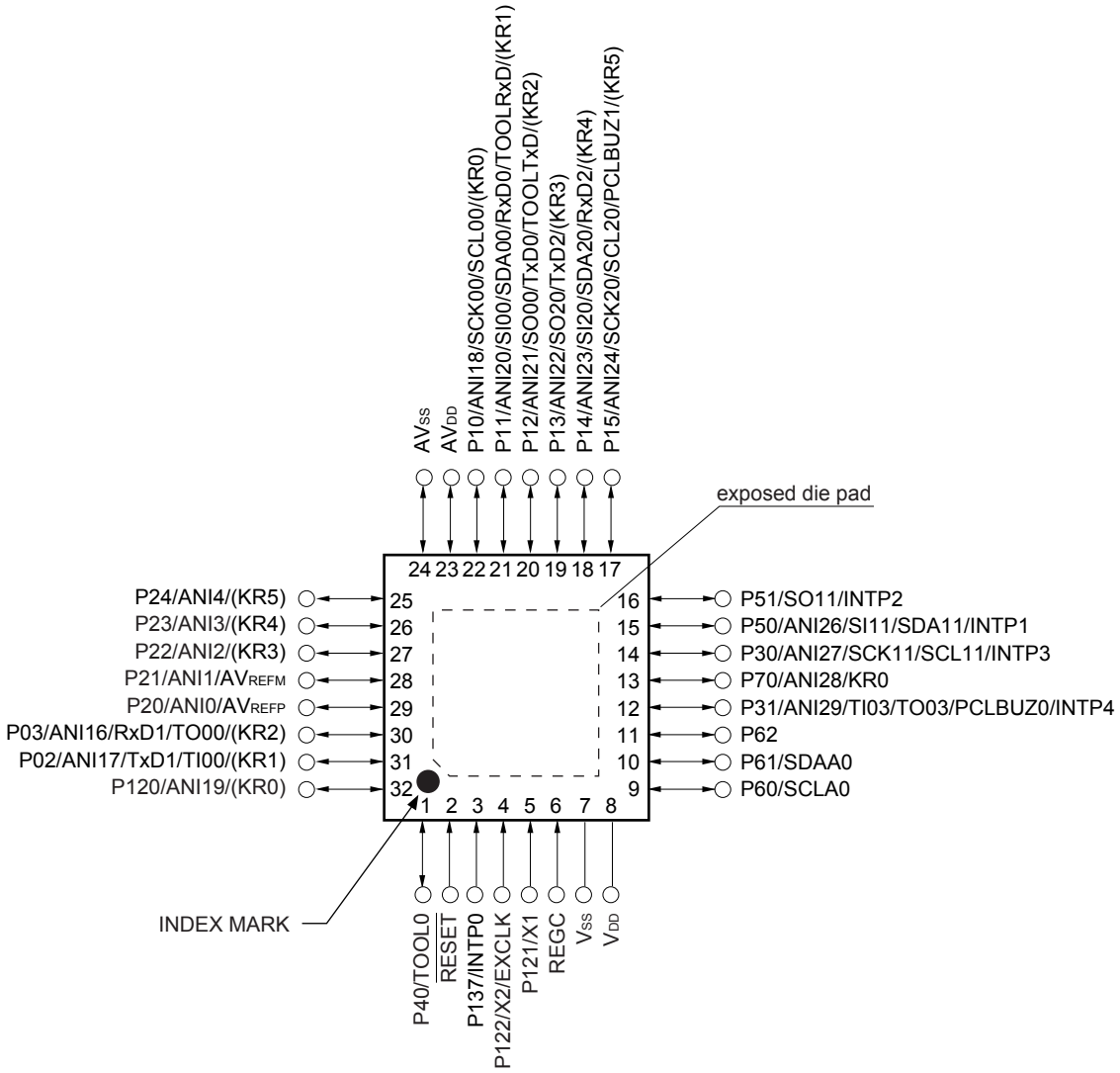
**Caution Connect the REGC pin to Vss via a capacitor (0.47 to 1 μF).**

**Remarks 1.** For pin identification, see **1.4 Pin Identification**.

**2.** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

### 1.3.2 32-pin products

- 32-pin plastic HWQFN (5 × 5 mm, 0.5 mm pitch)




**Caution Connect the REGC pin to Vss via a capacitor (0.47 to 1 μF).**

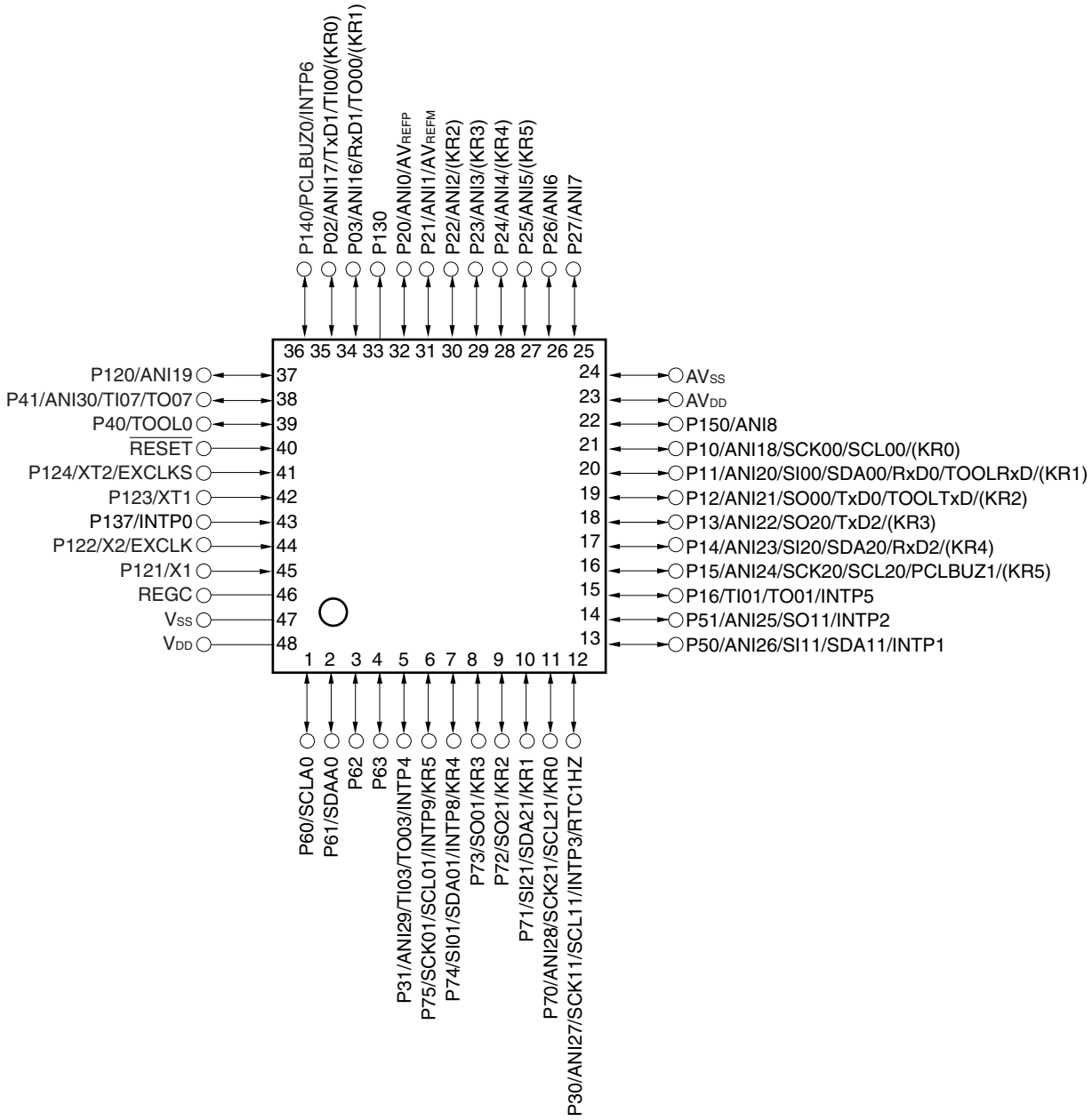
**Remarks 1.** For pin identification, see **1.4 Pin Identification**.

**2.** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

<R> **3.** It is recommended to connect an exposed die pad to Vss.

### 1.3.3 48-pin products

- 48-pin plastic LFQFP (7 × 7 mm, 0.5 mm pitch)

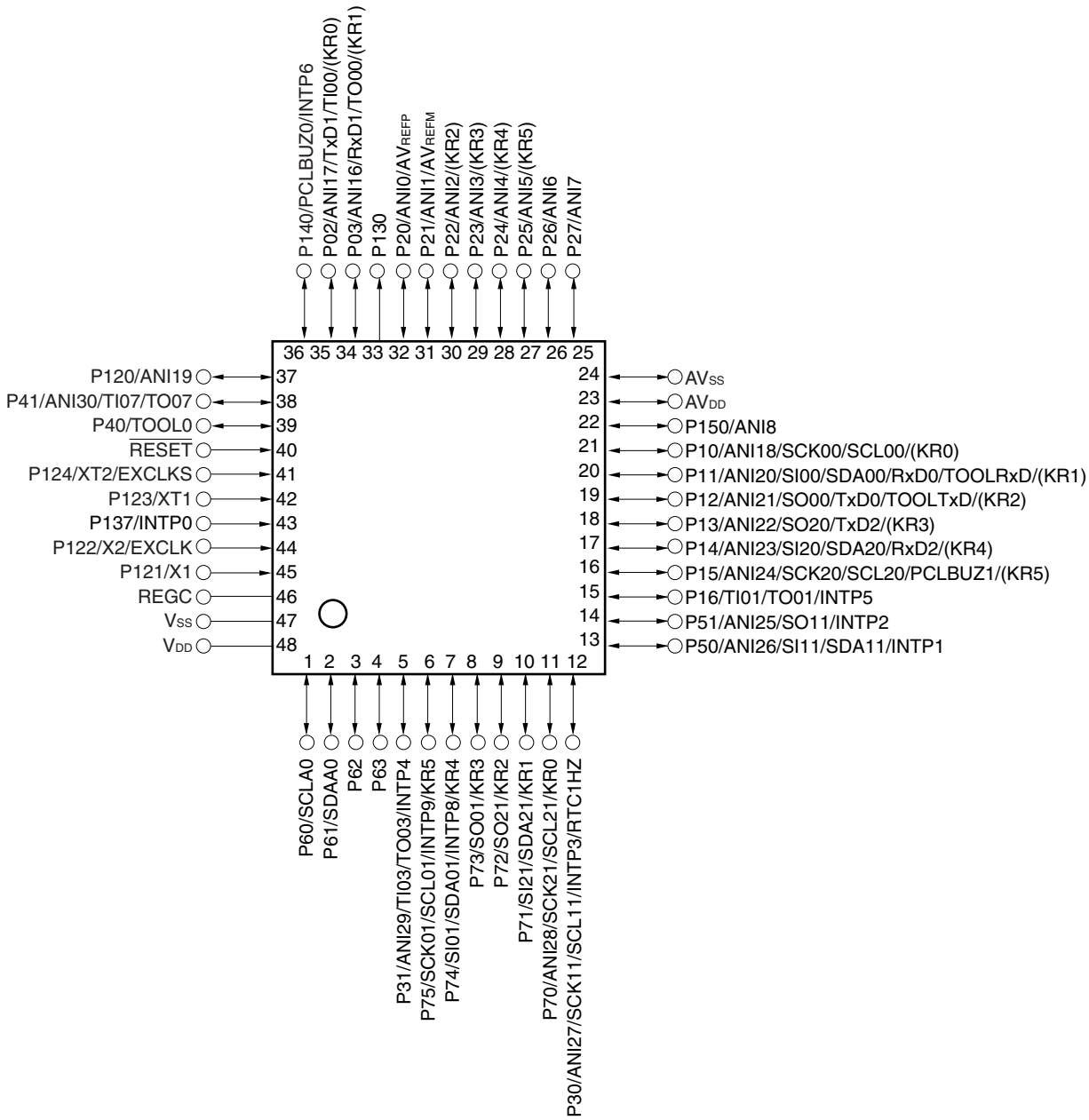



**Caution Connect the REGC pin to Vss via a capacitor (0.47 to 1 μF).**

**Remarks 1.** For pin identification, see **1.4 Pin Identification**.

**2.** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

- 48-pin plastic HWQFN (7 × 7 mm, 0.5 mm pitch)



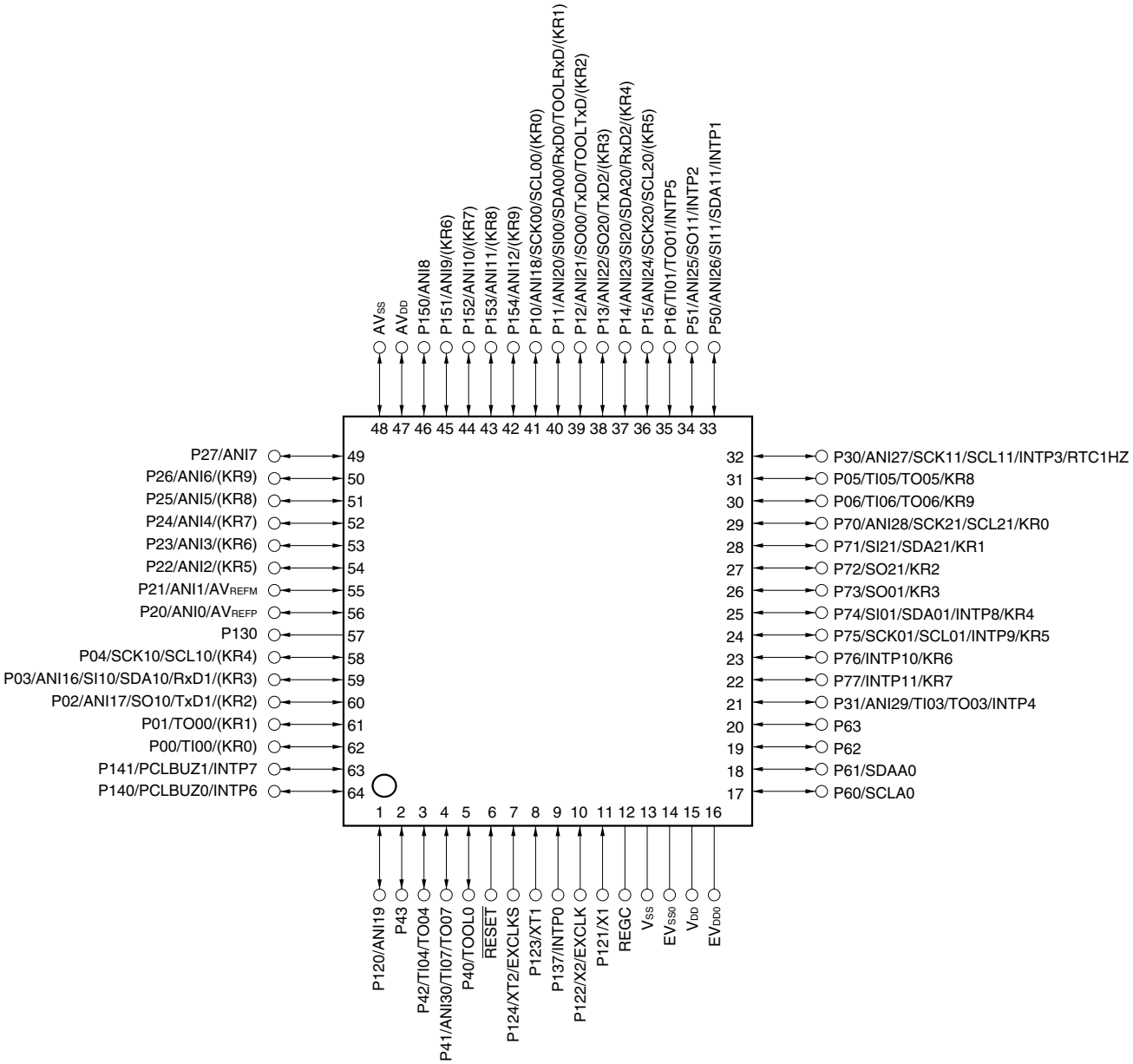

**Caution Connect the REGC pin to Vss via a capacitor (0.47 to 1 $\mu$F).**

**Remarks 1.** For pin identification, see **1.4 Pin Identification**.

**2.** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

### 1.3.4 64-pin products

- 64-pin plastic LFQFP (10 × 10 mm, 0.5 mm pitch)




**Cautions 1. Make $EV_{SS0}$ pin the same potential as $V_{SS}$ pin.**

**2. Make $V_{DD}$ pin the potential that is higher than $EV_{DD0}$ pin.**

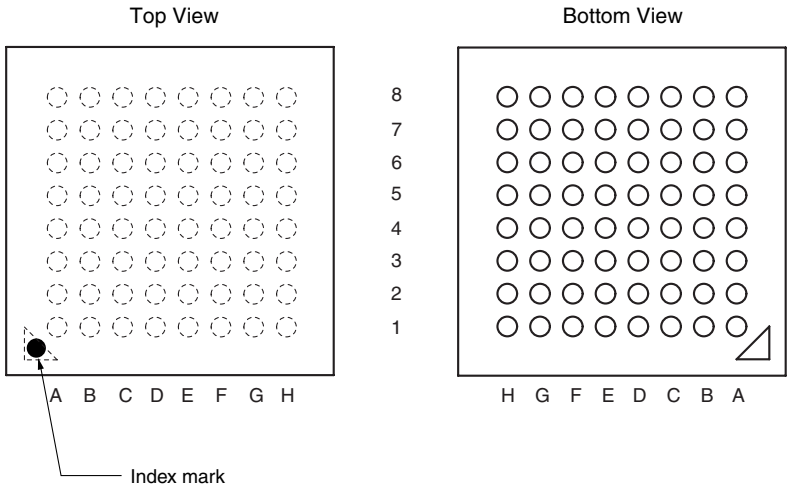
**3. Connect the REGC pin to Vss via a capacitor (0.47 to 1 μF).**

**Remarks 1.** For pin identification, see **1.4 Pin Identification.**

**2.** When using the microcontroller for an application where the noise generated inside the microcontroller must be reduced, it is recommended to supply separate powers to the $V_{DD}$ and $EV_{DD0}$ pins and connect the $V_{SS}$ and $EV_{SS}$0pins to separate ground lines.

**3.** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

- 64-pin plastic VFBGA (4 × 4 mm, 0.4 mm pitch)




| Pin No. | Name | Pin No. | Name | Pin No. | Name | Pin No. | Name |
|---|---|---|---|---|---|---|---|
| A1 | P05/TI05/TO05/KR8 | C1 | P51/ANI25/SO11 /INTP2 | E1 | P153/ANI11/(KR8) | G1 | $AV_{DD}$ |
| A2 | P30/ANI27/SCK11 /SCL11/INTP3 /RTC1HZ | C2 | P71/SI21/SDA21/KR1 | E2 | P154/ANI12/(KR9) | G2 | P25/ANI5/(KR8) |
| A3 | P70/ANI28/SCK21 /SCL21/KR0 | C3 | P74/SI01/SDA01 /INTP8/KR4 | E3 | P10/ANI18/SCK00 /SCL00/(KR0) | G3 | P24/ANI4/(KR7) |
| A4 | P75/SCK01/SCL01 /INTP9/KR5 | C4 | P16/TI01/TO01/INTP5 | E4 | P11/ANI20/SI00 /SDA00/RxD0 /TOOLRxD/(KR1) | G4 | P22/ANI2/(KR5) |
| A5 | P77/INTP11/KR7 | C5 | P15/ANI24/SCK20 /SCL20/(KR5) | E5 | P03/ANI16/SI10 /SDA10/RxD1/(KR3) | G5 | P130 |
| A6 | P61/SDAA0 | C6 | P63 | E6 | P41/ANI30/TI07/TO07 | G6 | P02/ANI17/SO10/TxD1 /(KR2) |
| A7 | P60/SCLA0 | C7 | $V_{SS}$ | E7 | $\overline{RESET}$ | G7 | P00/TI00/(KR0) |
| A8 | $EV_{DD0}$ | C8 | P121/X1 | E8 | P137/INTP0 | G8 | P124/XT2/EXCLKS |
| B1 | P50/ANI26 /SI11 /SDA11/INTP1 | D1 | P13/ANI22/SO20 /TxD2/(KR3) | F1 | P150/ANI8 | H1 | $AV_{SS}$ |
| B2 | P72/SO21/KR2 | D2 | P06/TI06/TO06/KR9 | F2 | P151/ANI9/(KR6) | H2 | P27/ANI7 |
| B3 | P73/SO01/KR3 | D3 | P12/ANI21/SO00 /TxD0/TOOLTxD/(KR2) | F3 | P152/ANI10/(KR7) | H3 | P26/ANI6/(KR9) |
| B4 | P76/INTP10/KR6 | D4 | P14/ANI23/SI20/ SDA20/RxD2/(KR4) | F4 | P21/ANI1/$AV_{REFM}$ | H4 | P23/ANI3/(KR6) |
| B5 | P31/ANI29/TI03/TO03 /INTP4 | D5 | P42/TI04/TO04 | F5 | P04/SCK10/SCL10 /(KR4) | H5 | P20/ANI0/$AV_{REFP}$ |
| B6 | P62 | D6 | P40/TOOL0 | F6 | P43 | H6 | P141/PCLBUZ1/INTP7 |
| B7 | $V_{DD}$ | D7 | REGC | F7 | P01/TO00/(KR1) | H7 | P140/PCLBUZ0/INTP6 |
| B8 | $EV_{SS0}$ | D8 | P122/X2/EXCLK | F8 | P123/XT1 | H8 | P120/ANI19 |

**Cautions 1. Make $EV_{SS0}$ pin the same potential as $V_{SS}$ pin.**

**2. Make $V_{DD}$ pin the potential that is higher than $EV_{DD0}$ pin.**

**3. Connect the REGC pin to Vss via a capacitor (0.47 to 1 μF).**

**Remarks 1.** For pin identification, see **1.4 Pin Identification.**

**2.** When using the microcontroller for an application where the noise generated inside the microcontroller must be reduced, it is recommended to supply separate powers to the $V_{DD}$ and $EV_{DD0}$ pins and connect the $V_{SS}$ and $EV_{SS0}$ pins to separate ground lines.

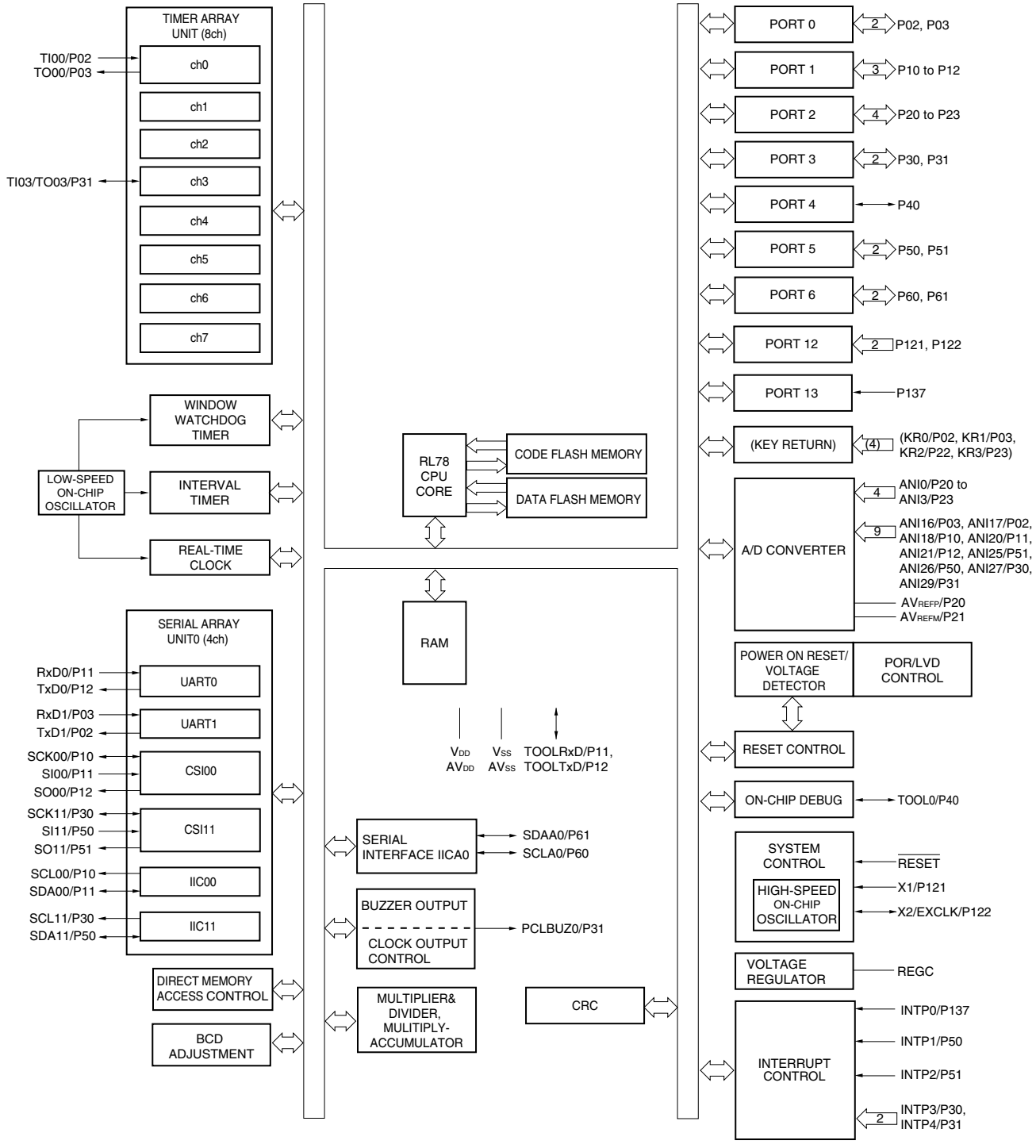
**3.** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

## 1.4 Pin Identification

| | |
|---|---|
| ANI0 to ANI12, ANI16 to ANI30: | Analog input |
| $AV_{DD}$: | Analog power supply |
| $AV_{SS}$: | Analog ground |
| $AV_{REFM}$: | A/D converter reference potential (– side) input |
| $AV_{REFP}$: | A/D converter reference potential (+ side) input |
| $EV_{DD0}$: | Power supply for port |
| $EV_{SS0}$: | Ground for port |
| EXCLK: | External clock input (main system clock) |
| EXCLKS: | External clock input (subsystem clock) |
| INTP0 to INTP11: | Interrupt Request from External |
| KR0 to KR9: | Key return |
| P00 to P06: | Port 0 |
| P10 to P16: | Port 1 |
| P20 to P27: | Port 2 |
| P30, P31: | Port 3 |
| P40 to P43: | Port 4 |
| P50, P51: | Port 5 |
| P60 to P63: | Port 6 |
| P70 to P77: | Port 7 |
| P120 to P124: | Port 12 |
| P130, P137: | Port 13 |
| P140, P141: | Port 14 |
| P150 to P154: | Port 15 |
| PCLBUZ0, PCLBUZ1: | Programmable clock output/buzzer output |
| REGC: | Regulator capacitance |
| $\overline{\text{RESET}}$: | Reset |
| RTC1HZ: | Real-time clock correction clock (1 Hz) output |
| RxD0 to RxD2: | Receive data |
| SCK00, SCK01, SCK10, SCK11, SCK20, SCK21: | Serial clock input/output |
| SCLA0, SCL00, SCL01, SCL10, SCL11, SCL20, SCL21: | Serial clock output |
| SDAA0, SDA00, SDA01, SDA10, SDA11, SDA20, SDA21: | Serial data input/output |
| SI00, SI01, SI10, SI11, SI20, SI21: | Serial data input |
| SO00, SO01, SO10, SO11, SO20, SO21: | Serial data output |
| TI00, TI01, TI03 to TI07: | Timer input |
| TO00, TO01, TO03 to TO07: | Timer output |
| TOOL0: | Data input/output for tool |
| TOOLRxD, TOOLTxD: | Data input/output for external device |
| TxD0 to TxD2: | Transmit data |
| $V_{DD}$: | Power supply |
| $V_{SS}$: | Ground |
| X1, X2: | Crystal oscillator (main system clock) |
| XT1, XT2: | Crystal oscillator (subsystem clock) |

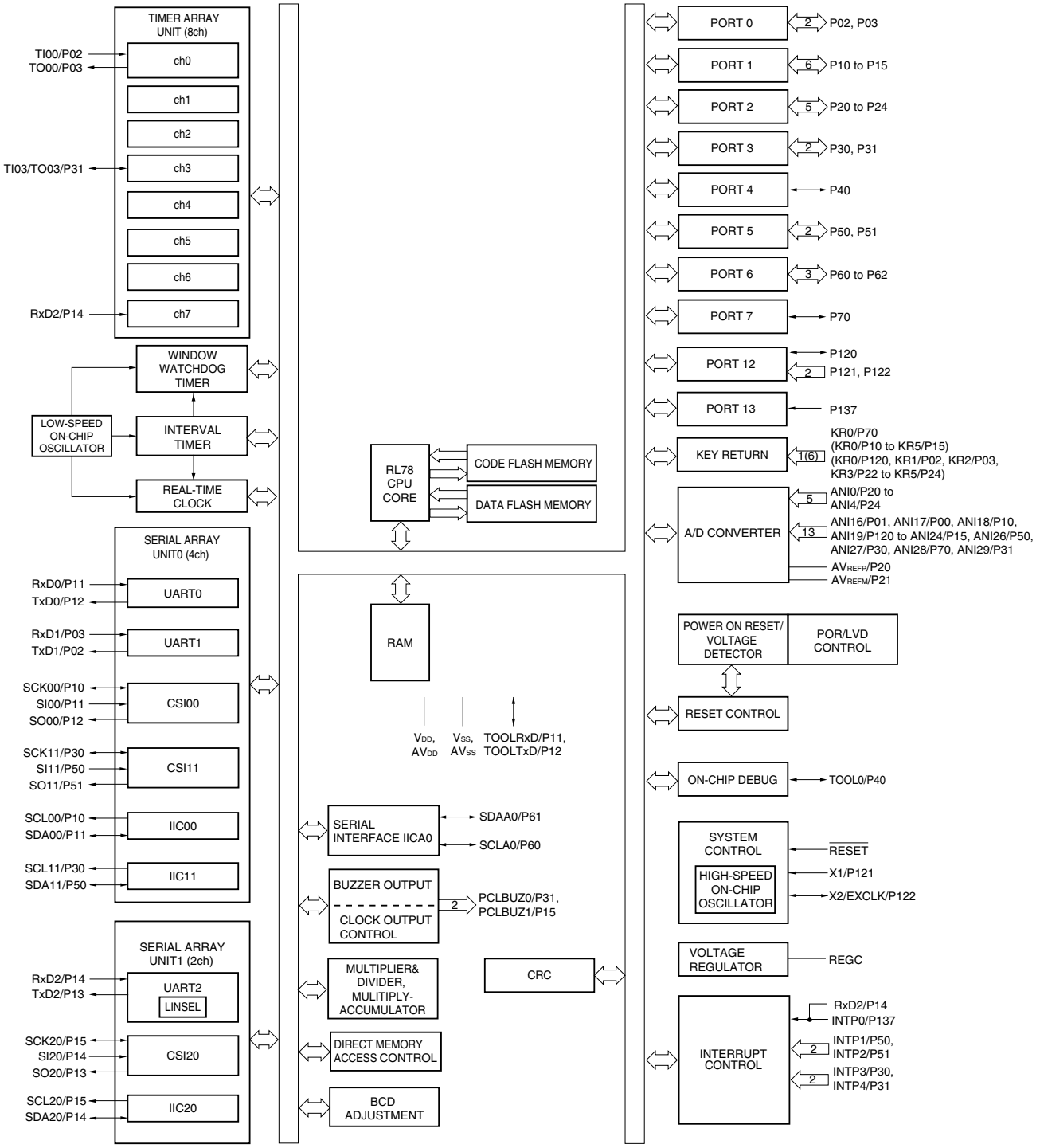
## 1.5 Block Diagram

### 1.5.1 25-pin products



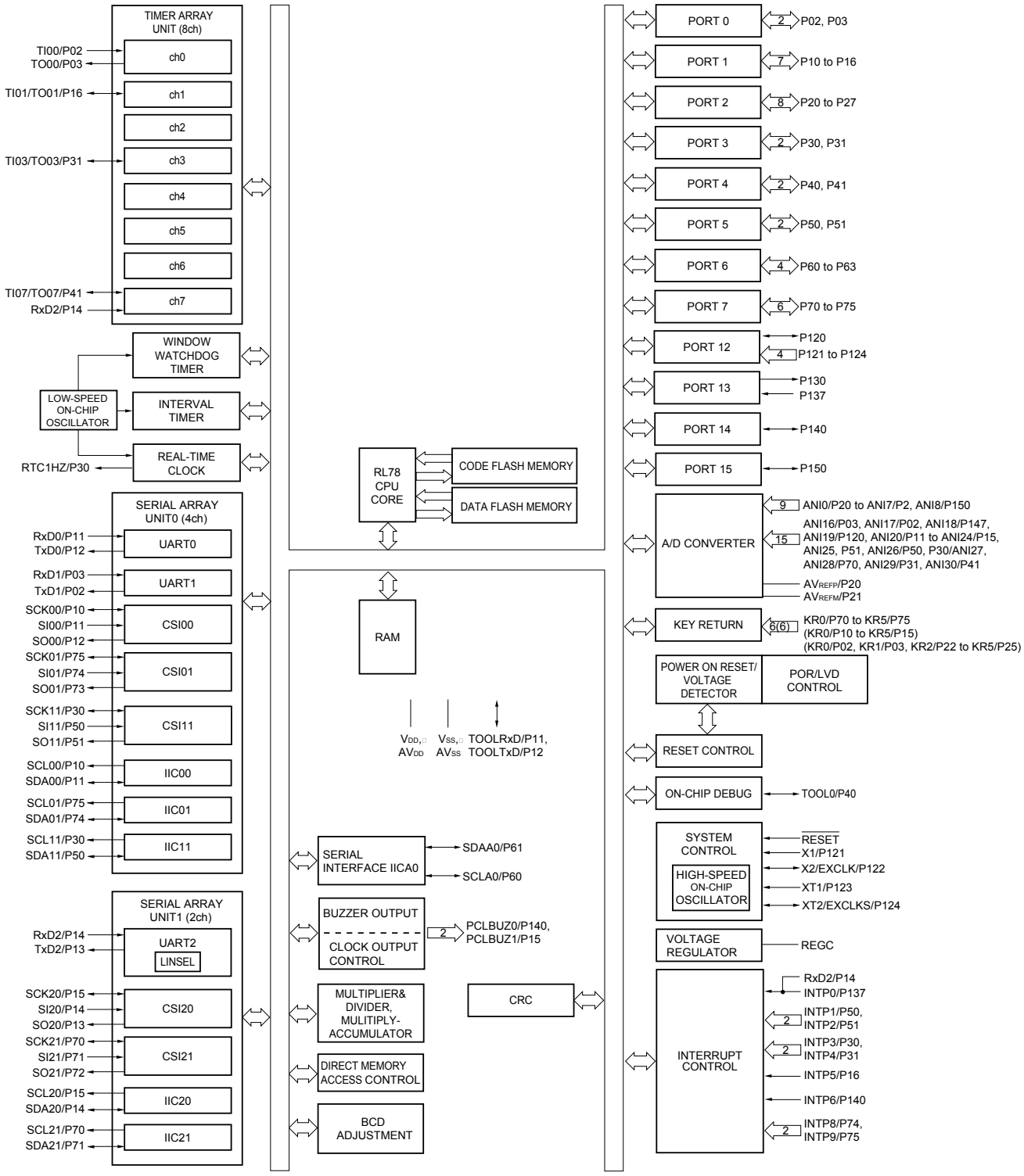

**Remark** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

### <R> 1.5.2 32-pin products



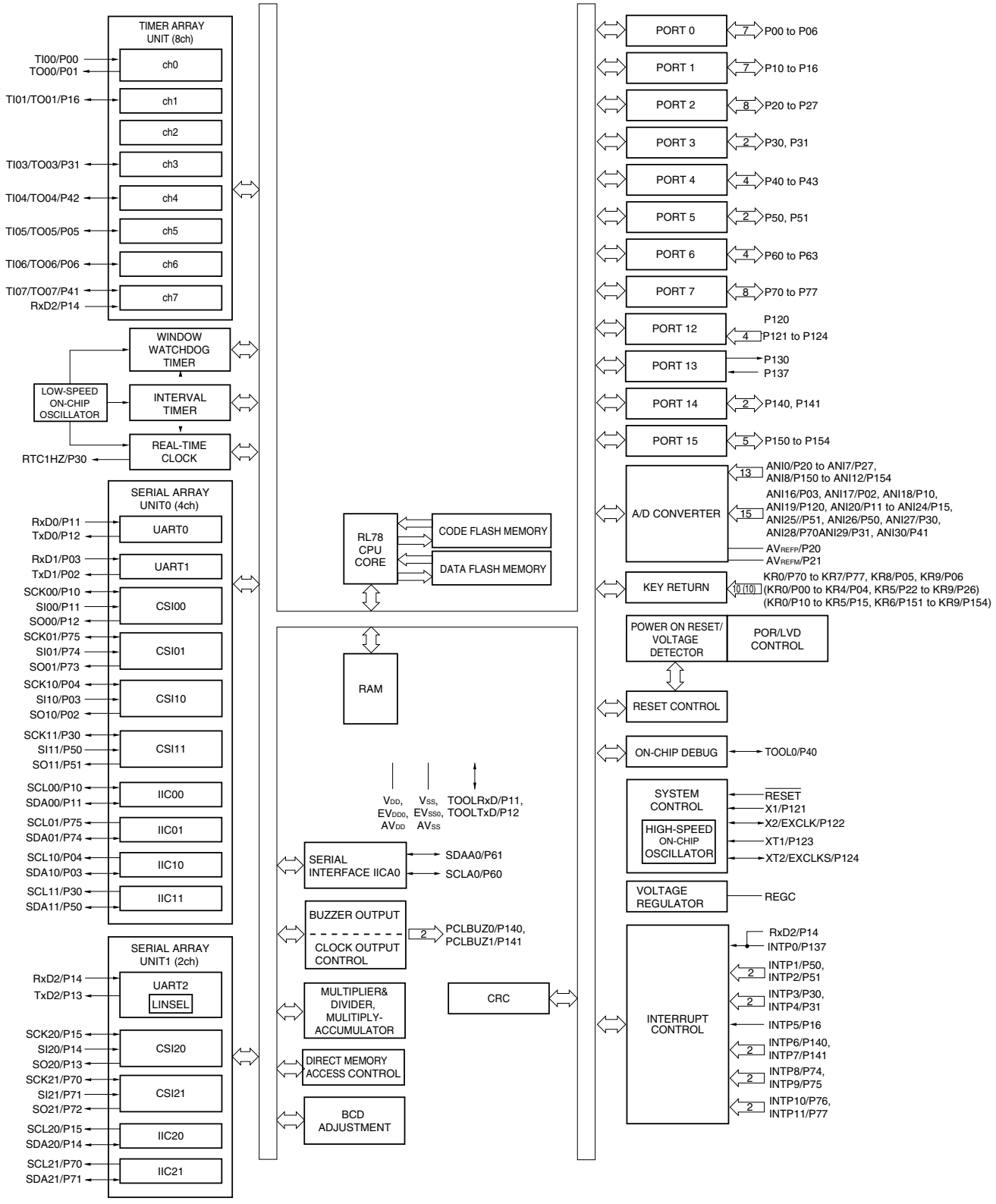

**Remark** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

### <R> 1.5.3 48-pin products




**Remark** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

### 1.5.4 64-pin products




**Remark** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

## 1.6 Outline of Functions

(1/2)

| Item | | 25-pin | 32-pin | 48-pin | 64-pin |
|---|---|---|---|---|---|
| | | R5F10E8x | R5F10EBx | R5F10EGx | R5F10ELx |
| Code flash memory (KB) | | 16 to 64 | 16 to 64 | 16 to 64 | 32 to 64 |
| Data flash memory (KB) | | 4 | 4 | 4 | 4 |
| RAM (KB) | | 2 to 4 Note1 | 2 to 4 Note1 | 2 to 4 Note1 | 2 to 4 Note1 |
| Address space | | 1 MB | | | |
| Main system clock | High-speed system clock | X1 (crystal/ceramic) oscillation, external main system clock input (EXCLK)<br>1 to 20 MHz: $V_{DD}$ = 2.7 to 3.6 V, 1 to 8 MHz: $V_{DD}$ = 1.8 to 2.7 V, 1 to 4 MHz: $V_{DD}$ = 1.6 to 1.8 V | | | |
| | High-speed on-chip oscillator | HS (High-speed main) mode : 1 to 32 MHz ($V_{DD}$ = 2.7 to 3.6 V),<br>HS (High-speed main) mode : 1 to 16 MHz ($V_{DD}$ = 2.4 to 3.6 V),<br>LS (Low-speed main) mode : 1 to 8 MHz ($V_{DD}$ = 1.8 to 3.6 V),<br>LV (Low-voltage main) mode : 1 to 4 MHz ($V_{DD}$ = 1.6 to 3.6 V) | | | |
| Subsystem clock | | – | | XT1 (crystal) oscillation, external subsystem clock input (EXCLKS) 32.768 kHz (TYP.) | |
| Low-speed on-chip oscillator | | 15 kHz (TYP.) | | | |
| General-purpose register | | (8-bit register × 8) × 4 bank | | | |
| Minimum instruction execution time | | 0.03125 μs (High-speed on-chip oscillator: $f_{IH}$ = 32 MHz operation) | | | |
| | | 0.05 μs (High-speed system clock: $f_{MX}$ = 20 MHz operation) | | | |
| | | – | | 30.5 μs (Subsystem clock: $f_{SUB}$ = 32.768 kHz operation) | |
| Instruction set | | • Data transfer (8/16 bits)<br>• Adder and subtractor/logical operation (8/16 bits)<br>• Multiplication (8 bits × 8 bits)<br>• Rotate, barrel shift, and bit manipulation (Set, reset, test, and Boolean operation), etc. | | | |
| I/O port | Total | 19 | 26 | 42 | 56 |
| <R> | CMOS I/O | 14<br>(N-ch O.D. I/O [$V_{DD}$ withstand voltage]: 6) | 20<br>(N-ch O.D. I/O [$V_{DD}$ withstand voltage]: 9) | 32<br>(N-ch O.D. I/O [$V_{DD}$ withstand voltage]: 11) | 46<br>(N-ch O.D. I/O [$V_{DD}$ withstand voltage]: 12) |
| | CMOS input | 3 | 3 | 5 | 5 |
| | CMOS output | – | – | 1 | 1 |
| | N-ch open-drain I/O (6 V tolerance) | 2 | 3 | 4 | 4 |
| Timer | 16-bit timer | 8 channels | | | |
| | Watchdog timer | 1 channel | | | |
| | Real-time clock (RTC) | 1 channel Note 2 | | 1 channel | |
| | 12-bit interval timer (IT) | 1 channel | | | |
| | Timer output | 2 channels (PWM outputs: 1 Note 3) | | 4 channels<br>(PWM outputs: 3 Note 3) | 7 channels<br>(PWM outputs: 6 Note 3) |
| | RTC output | – | | 1<br>• 1 Hz (subsystem clock: $f_{SUB}$ = 32.768 kHz) | |

**Notes 1.** In the case of the 4 KB, this is about 3 KB when the self-programming function and data flash function are used. (For details, see **CHAPTER 3**)

<R> **2.** Only the constant-period interrupt function when the low-speed on-chip oscillator clock ($f_{IL}$) is selected.

**3.** The number of PWM outputs varies depending on the setting of channels in use (the number of masters and slaves). (**6.9.3 Operation as multiple PWM output function**).

(2/2)

| Item | | 25-pin<br>R5F10E8x | 32-pin<br>R5F10EBx | 48-pin<br>R5F10EGx | 64-pin<br>R5F10ELx |
|---|---|---|---|---|---|
| Clock output/buzzer output | | 1 | 2 | 2 | 2 |
| | | • 2.44 kHz, 4.88 kHz, 9.76 kHz, 1.25 MHz, 2.5 MHz, 5 MHz, 10 MHz (Main system clock: $f_{MAIN}$ = 20 MHz operation) | | • 2.44 kHz, 4.88 kHz, 9.76 kHz, 1.25 MHz, 2.5 MHz, 5 MHz, 10 MHz (Main system clock: $f_{MAIN}$ = 20 MHz operation)<br>• 256 Hz, 512 Hz, 1.024 kHz, 2.048 kHz, 4.096 kHz, 8.192 kHz, 16.384 kHz, 32.768 kHz (Subsystem clock: $f_{SUB}$ = 32.768 kHz operation) | |
| 8/12-bit resolution A/D converter | | 13 channels | 18 channels | 24 channels | 28 channels |
| Serial interface | | [25-pin products]<br>• CSI: 1 channel/simplified $I^2C$: 1 channel/UART: 1 channel<br>• CSI: 1 channel/simplified $I^2C$: 1 channel/UART: 1 channel<br>[32-pin products]<br>• CSI: 1 channel/simplified $I^2C$: 1 channel/UART: 1 channel<br>• CSI: 1 channel/simplified $I^2C$: 1 channel/UART: 1 channel<br>• CSI: 1 channel/simplified $I^2C$: 1 channel/UART (UART supporting LIN-bus): 1 channel<br>[48-pin products]<br>• CSI: 2 channels/simplified $I^2C$: 2 channels/UART: 1 channel<br>• CSI: 1 channel/simplified $I^2C$: 1 channel/UART: 1 channel<br>• CSI: 2 channels/simplified $I^2C$: 2 channels/UART (UART supporting LIN-bus): 1 channel<br>[64-pin products]<br>• CSI: 2 channels/simplified $I^2C$: 2 channels/UART: 1 channel<br>• CSI: 2 channels/simplified $I^2C$: 2 channels/UART: 1 channel<br>• CSI: 2 channels/simplified $I^2C$: 2 channels/UART (UART supporting LIN-bus): 1 channel | | | |
| | $I^2C$ bus | 1 channel | 1 channel | 1 channel | 1 channel |
| Multiplier and divider/multiply-accumulator | | • 16 bits × 16 bits = 32 bits (Unsigned or signed)<br>• 32 bits ÷ 32 bits = 32 bits (Unsigned)<br>• 16 bits × 16 bits + 32 bits = 32 bits (Unsigned or signed) | | | |
| DMA controller | | 2 channels | | | |
| Vectored interrupt sources | Internal | 24 | 27 | 27 | 27 |
| | External | 6 | 6 | 10 | 13 |
| Key interrupt | | 0 ch (4 ch)[Note 1] | 1 ch (6 ch)[Note 1] | 6 ch | 10 ch |
| Reset | | • Reset by $\overline{\text{RESET}}$ pin<br>• Internal reset by watchdog timer<br>• Internal reset by power-on-reset<br>• Internal reset by voltage detector<br>• Internal reset by illegal instruction execution[Note 2]<br>• Internal reset by RAM parity error<br>• Internal reset by illegal-memory access | | | |
| Power-on-reset circuit | | • Power-on-reset: 1.51 V (TYP.)<br>• Power-down-reset: 1.50 V (TYP.) | | | |
| Voltage detector | | • Rising edge : 1.67 V to 3.14 V (12 stages)<br>• Falling edge : 1.63 V to 3.06 V (12 stages) | | | |
| On-chip debug function | | Provided | | | |
| Power supply voltage | | $V_{DD}$ = 1.6 to 3.6 V | | | |
| Operating ambient temperature | | $T_A$ = −40 to +85°C (A: Consumer application), $T_A$ = −40 to +105°C (G: Industrial application) | | | |

**Notes 1.** Can be used by the Peripheral I/O redirection register (PIOR).

**2.** The illegal instruction is generated when instruction code FFH is executed.
Reset by the illegal instruction execution not issued by emulation with the in-circuit emulator or on-chip debug emulator.

# CHAPTER 2 PIN FUNCTIONS

## 2.1 Port Function

Pin I/O buffer power supplies depend on the product. The relationship between these power supplies and the pins is shown below.

**Table 2-1. Pin I/O Buffer Power Supplies**

**(1) 25-pin products**

| Power Supply | Corresponding Pins |
|---|---|
| $V_{DD}$ | Port pins other than P20 to P23<br>• $\overline{\text{RESET}}$, REGC |
| $AV_{DD}$ | • P20 to P23 |

**(2) 32-pin products**

| Power Supply | Corresponding Pins |
|---|---|
| $V_{DD}$ | Port pins other than P20 to P24<br>• $\overline{\text{RESET}}$, REGC |
| $AV_{DD}$ | • P20 to P24 |

**(3) 48-pin products**

| Power Supply | Corresponding Pins |
|---|---|
| $V_{DD}$ | Port pins other than P20 to P27, P150<br>• $\overline{\text{RESET}}$, REGC |
| $AV_{DD}$ | • P20 to P27, P150 |

**(4) 64-pin products**

| Power Supply | Corresponding Pins |
|---|---|
| $EV_{DD0}$ | Port pins other than P20 to P27, P121 to P124, P137, and P150 to P154 |
| $V_{DD}$ | • P121 to P124, P137<br>• $\overline{\text{RESET}}$, REGC |
| $AV_{DD}$ | • P20 to P27, and P150 to P154 |

Set in each port I/O, buffer, pull-up resistor is also valid for alternate functions.

### 2.1.1 25-pin products

(1/2)

<R>

| Function Name | Pin Type | I/O | After Reset | Alternate Function | Function |
|---|---|---|---|---|---|
| P02 | 7-3-2 | I/O | Analog input port | ANI17/TI00/TxD1/ (KR0) | Port 0.<br>2-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Input of P03 can be set to TTL input buffer.<br>Output of P02 and P03 can be set to N-ch open-drain output ($V_{DD}$ tolerance).<br>Can be set to analog input[Note 1]. |
| P03 | 8-3-2 | | | ANI16/TO00/RxD1/ (KR1) | |
| P10 | 8-3-2 | I/O | Analog input port | ANI18/SCK00/SCL00 | Port 1.<br>3-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Input of P10 and P11 can be set to TTL input buffer.<br>Output of P10 to P12 can be set to N-ch open-drain output ($V_{DD}$ tolerance).<br>Can be set to analog input[Note 1]. |
| P11 | | | | ANI20/SI00/RxD0/ TOOLRxD/SDA00 | |
| P12 | 7-3-2 | | | ANI21/SO00/TxD0/ TOOLTxD | |
| P20 | 4-3-1 | I/O | Analog input port | ANI0/$AV_{REFP}$ | Port 2.<br>4-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Can be set to analog input[Note 2]. |
| P21 | | | | ANI1/$AV_{REFM}$ | |
| P22 | | | | ANI2/(KR2) | |
| P23 | | | | ANI3/(KR3) | |
| P30 | 7-3-1 | I/O | Analog input port | ANI27/INTP3/ SCK11/SCL11 | Port 3.<br>2-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Can be set to analog input[Note 1]. |
| P31 | | | | ANI29/TI03/TO03/ INTP4/PCLBUZ0 | |
| P40 | 7-1-1 | I/O | Input port | TOOL0 | Port 4.<br>1-bit I/O port.<br>Input/output can be specified.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port. |
| P50 | 7-3-2 | I/O | Analog input port | ANI26/INTP1/SI11/ SDA11 | Port 5.<br>2-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Output of P50 can be set to N-ch open-drain output ($V_{DD}$ tolerance).<br>Can be set to analog input[Note 1]. |
| P51 | 7-3-1 | | | ANI25/INTP2/SO11 | |
| P60 | 12-1-1 | I/O | Input port | SCLA0 | Port 6.<br>2-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Output of P60 and P61 can be set to N-ch open-drain output (6 V tolerance). |
| P61 | | | | SDAA0 | |

**Notes 1.** Digital or analog for each pin can be selected with the port mode control register x (PMCx) (can be set in 1-bit units).

**2.** Digital or analog for each pin can be selected with the A/D port configuration register (ADPC).

**Remark** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

(2/2)

<R>

| Function Name | Pin Type | I/O | After Reset | Alternate Function | Function |
|---|---|---|---|---|---|
| P121 | 2-2-1 | Input | Input port | X1 | Port 12.<br>2-bit input only port. |
| P122 | | | | X2/EXCLK | |
| P137 | 2-1-2 | Input | Input port | INTP0 | Port 13.<br>1-bit input only port. |
| $\overline{\text{RESET}}$ | 2-1-1 | Input | – | – | Input only pin for external reset<br>When external reset is not used, connect this pin to $V_{DD}$ directly or via a resistor. |

### 2.1.2 32-pin products

(1/2)

<R>

| Function Name | Pin Type | I/O | After Reset | Alternate Function | Function |
|---|---|---|---|---|---|
| P02 | 7-3-2 | I/O | Analog input port | ANI17/TI00/TxD1/ (KR1) | Port 0.<br>2-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Input of P03 can be set to TTL input buffer.<br>Output of P02 and P03 can be set to N-ch open-drain output ($V_{DD}$ tolerance).<br>Can be set to analog input[Note 1]. |
| P03 | 8-3-2 | | | ANI16/TO00/RxD1/ (KR2) | |
| P10 | 8-3-2 | I/O | Analog input port | ANI18/SCK00/ SCL00/(KR0) | Port 1.<br>6-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Input of P10, P11, P14, and P15 can be set to TTL input buffer.<br>Output of P10 to P15 can be set to N-ch open-drain output ($V_{DD}$ tolerance).<br>Can be set to analog input. |
| P11 | | | | ANI20/SI00/RxD0/ TOOLRxD/SDA00/ (KR1) | |
| P12 | 7-3-2 | | | ANI21/SO00/TxD0/ TOOLTxD/(KR2) | |
| P13 | | | | ANI22/TxD2/SO20/ (KR3) | |
| P14 | 8-3-2 | | | ANI23/RxD2/SI20/ SDA20/(KR4) | |
| P15 | | | | ANI24/PCLBUZ1/ SCK20/SCL20/(KR5) | |
| P20 | 4-3-1 | I/O | Analog input port | ANI0/$AV_{REFP}$ | Port 2.<br>5-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Can be set to analog input[Note 2]. |
| P21 | | | | ANI1/$AV_{REFM}$ | |
| P22 | | | | ANI2/(KR3) | |
| P23 | | | | ANI3/(KR4) | |
| P24 | | | | ANI4/(KR5) | |
| P30 | 7-3-1 | I/O | Analog input port | ANI27/INTP3/ SCK11/SCL11 | Port 3.<br>2-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Can be set to analog input[Note 1]. |
| P31 | | | | ANI29/TI03/TO03/ INTP4/PCLBUZ0 | |
| P40 | 7-1-1 | I/O | Input port | TOOL0 | Port 4.<br>1-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port. |

**Notes 1.** Digital or analog for each pin can be selected with the port mode control register x (PMCx) (can be set in 1-bit units.

**2.** Digital or analog for each pin can be selected with the A/D port configuration register (ADPC).

**Remark** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

(2/2)

<R>

| Function Name | Pin Type | I/O | After Reset | Alternate Function | Function |
|---|---|---|---|---|---|
| P50 | 7-3-2 | I/O | Analog input port | ANI26/INTP1/SI11/ SDA11 | Port 5.<br>2-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Output of P50 can be set to N-ch open-drain output ($V_{DD}$ tolerance).<br>P50 can be set to analog input[Note] |
| P51 | 7-1-1 | | Input port | INTP2/SO11 | |
| P60 | 12-1-1 | I/O | Input port | SCLA0 | Port 6.<br>3-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Output of P60 to P62 can be set to N-ch open-drain output (6 V tolerance). |
| P61 | | | | SDAA0 | |
| P62 | | | | – | |
| P70 | 7-3-1 | I/O | Analog input port | ANI28/KR0 | Port 7.<br>1-bit I/O port.<br>Input/output can be specified.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Can be set to analog input[Note]. |
| P120 | 7-3-1 | I/O | Analog input port | ANI19/(KR0) | Port 12.<br>1-bit I/O port and 2-bit input only port.<br>P120 can be set to analog input[Note].<br>For only P120, input/output can be specified.<br>For only P120, use of an on-chip pull-up resistor can be specified by a software setting at input port. |
| P121 | 2-2-1 | Input | Input port | X1 | |
| P122 | | | | X2/EXCLK | |
| P137 | 2-1-2 | Input | Input port | INTP0 | Port 13<br>1bit input only port. |
| $\overline{\text{RESET}}$ | 2-1-1 | Input | – | – | Input only pin for external reset<br>When external reset is not used, connect this pin to $V_{DD}$ directly or via a resistor. |

**Note** Digital or analog for each pin can be selected with the port mode control register x (PMCx) (can be set in 1-bit units).

**Remark** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

### 2.1.3 48-pin products

(1/2)

<R>

| Function Name | Pin Type | I/O | After Reset | Alternate Function | Function |
|---|---|---|---|---|---|
| P02 | 7-3-2 | I/O | Analog input port | ANI17/TI00/TxD1/ (KR0) | Port 0.<br>2-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Input of P03 can be set to TTL input buffer.<br>Output of P02 and P03 can be set to N-ch open-drain output ($V_{DD}$ tolerance).<br>Can be set to analog input[Note 1]. |
| P03 | 8-3-2 | | | ANI16TO00/RxD1/ (KR1) | |
| P10 | 8-3-2 | I/O | Analog input port | ANI18/SCK00/ SCL00/(KR0) | Port 1.<br>7-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Input of P10, P11, and P14 to P16 can be set to TTL input buffer.<br>Output of P10 to P15 can be set to N-ch open-drain output ($V_{DD}$ tolerance).<br>P10 to P15 can be set to analog input[Note 1]. |
| P11 | | | | ANI20/SI00/RxD0/ TOOLRxD/SDA00/ (KR1) | |
| P12 | 7-3-2 | | | ANI21/SO00/TxD0/ TOOLTxD/(KR2) | |
| P13 | | | | ANI22/TxD2/SO20/ (KR3) | |
| P14 | 8-3-2 | | | ANI23/RxD2/SI20/ SDA20/(KR4) | |
| P15 | | | | ANI24/PCLBUZ1/ SCK20/SCL20/(KR5) | |
| P16 | 8-1-1 | | Input port | TI01/TO01/INTP5 | |
| P20 | 4-3-1 | I/O | Analog input port | ANI0/$AV_{REFP}$ | Port 2.<br>8-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Can be set to analog input[Note 2]. |
| P21 | | | | ANI1/$AV_{REFM}$ | |
| P22 | | | | ANI2/(KR2) | |
| P23 | | | | ANI3/(KR3) | |
| P24 | | | | ANI4/(KR4) | |
| P25 | | | | ANI5/(KR5) | |
| P26 | | | | ANI6 | |
| P27 | | | | ANI7 | |
| P30 | 7-3-1 | I/O | Analog input port | ANI27/INTP3/ RTC1HZ/SCK11/ SCL11 | Port 3.<br>2-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Can be set to analog input[Note 1]. |
| P31 | | | | ANI29/TI03/TO03/ INTP4 | |

**Notes 1.** Digital or analog for each pin can be selected with the port mode control register x (PMCx) (can be set in 1-bit units).

**2.** Digital or analog for each pin can be selected with the A/D port configuration register (ADPC).

**Remark** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

(2/2)

<R>

| Function Name | Pin Type | I/O | After Reset | Alternate Function | Function |
|---|---|---|---|---|---|
| P40 | 7-1-1 | I/O | Input port | TOOL0 | Port 4.<br>2-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>P41 can be set to analog input[Note 1]. |
| P41 | 7-3-1 | | Analog input port | ANI30/TI07/TO07 | |
| P50 | 7-3-2 | I/O | Analog input port | ANI26/INTP1/SI11/SDA11 | Port 5.<br>2-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Output of P50 can be set to N-ch open-drain output ($V_{DD}$ tolerance).<br>Can be set to analog input[Note 1]. |
| P51 | 7-3-1 | | | ANI25/INTP2/SO11 | |
| P60 | 12-1-1 | I/O | Input port | SCLA0 | Port 6.<br>4-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>N-ch open-drain output (6 V tolerance). |
| P61 | | | | SDAA0 | |
| P62 | | | | – | |
| P63 | | | | – | |
| P70 | 7-3-1 | I/O | Analog input port | ANI28/KR0/SCK21/SCL21 | Port 7.<br>6-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Output of P71 and P74 can be set to N-ch open-drain output ($V_{DD}$ tolerance).<br>P70 can be set to analog input[Note 1]. |
| P71 | 7-1-2 | | Input port | KR1/SI21/SDA21 | |
| P72 | 7-1-1 | | | KR2/SO21 | |
| P73 | | | | KR3/SO01 | |
| P74 | 7-1-2 | | | KR4/INTP8/SI01/SDA01 | |
| P75 | 7-1-1 | | | KR5/INTP9/SCK01/SCL01 | |
| P120 | 7-3-1 | I/O | Analog input port | ANI19 | Port 12.<br>1-bit I/O port and 4-bit input only port.<br>For only P120, input/output can be specified.<br>For only P120, use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>P120 can be set to analog input[Note 1]. |
| P121 | 2-2-1 | Input | Input port | X1 | |
| P122 | | | | X2/EXCLK | |
| P123 | | | | XT1 | |
| P124 | | | | XT2/EXCLKS | |
| P130 | 1-1-1 | Output | Output port | – | Port 13.<br>1-bit output port and 1-bit input only port. |
| P137 | 2-1-2 | Input | Input port | INTP0 | |
| P140 | 7-1-1 | I/O | Input port | PCLBUZ0/INTP6 | Port 14.<br>1-bit I/O port.<br>Input/output can be specified.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port. |
| P150 | 4-3-1 | | Analog input port | ANI18 | Port 15<br>1-bit I/O port.<br>Input/output can be specified.<br>Can be set to analog input[Note 2]. |
| $\overline{\text{RESET}}$ | 2-1-1 | Input | – | – | Input only pin for external reset<br>When external reset is not used, connect this pin to $V_{DD}$ directly or via a resistor. |

**Notes 1.** Digital or analog for each pin can be selected with the port mode control register x (PMCx) (can be set in 1-bit units).

**2.** Digital or analog for each pin can be selected with the A/D port configuration register (ADPC).

### 2.1.4 64-pin products

(1/3)

<R>

| Function Name | Pin Type | I/O | After Reset | Alternate Function | Function |
|---|---|---|---|---|---|
| P00 | 8-1-1 | I/O | Input port | TI00/(KR0) | Port 0.<br>7-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Input of P00, P01, P03, and P04 can be set to TTL input buffer.<br>Output of P02 to P04 can be set to N-ch open-drain output ($V_{DD}$ tolerance).<br>P02 and P03 can be set to analog input[Note 1]. |
| P01 | | | | TO00/(KR1) | |
| P02 | 7-3-2 | | Analog input port | ANI17/SO10/TxD1/(KR2) | |
| P03 | 8-3-2 | | | ANI16/SI10/RxD1/SDA10/(KR3) | |
| P04 | 8-1-2 | | Input port | SCK10/SCL10/(KR4) | |
| P05 | 7-1-1 | | | TI05/TO05/KR8 | |
| P06 | | | | TI06/TO06/KR9 | |
| P10 | 8-3-2 | I/O | Analog input port | ANI18/SCK00/SCL00/(KR0) | Port 1.<br>7-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Input of P10, P11, and P14 to P16 can be set to TTL input buffer.<br>Output of P10 to P15 can be set to N-ch open-drain output ($V_{DD}$ tolerance).<br>P10 to P15 can be set to analog input[Note 1]. |
| P11 | | | | ANI20/SI00/RxD0/TOOLRxD/SDA00/(KR1) | |
| P12 | 7-3-2 | | | ANI21/SO00/TxD0/TOOLTxD/(KR2) | |
| P13 | | | | ANI22/TxD2/SO20/(KR3) | |
| P14 | 8-3-2 | | | ANI23/RxD2/SI20/SDA20/(KR4) | |
| P15 | | | | ANI24/SCK20/SCL20/(KR5) | |
| P16 | 8-1-1 | | Input port | TI01/TO01/INTP5 | |
| P20 | 4-3-1 | I/O | Analog input port | ANI0/$AV_{REFP}$ | Port 2.<br>8-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Can be set to analog input[Note 2]. |
| P21 | | | | ANI1/$AV_{REFM}$ | |
| P22 | | | | ANI2/(KR5) | |
| P23 | | | | ANI3/(KR6) | |
| P24 | | | | ANI4/(KR7) | |
| P25 | | | | ANI5/(KR8) | |
| P26 | | | | ANI6/(KR9) | |
| P27 | | | | ANI7 | |
| P30 | 7-3-2 | I/O | Analog input port | ANI27/INTP3/RTC1HZ/SCK11/SCL11 | Port 3.<br>2-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Can be set to analog input[Note 1]. |
| P31 | | | | ANI29/TI03/TO03/INTP4 | |

**Notes 1.** Digital or analog for each pin can be selected with the port mode control register x (PMCx) (can be set in 1-bit units).

**2.** Digital or analog for each pin can be selected with the A/D port configuration register (ADPC).

**Remark** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

(2/3)

<R>

| Function Name | Pin Type | I/O | After Reset | Alternate Function | Function |
|---|---|---|---|---|---|
| P40 | 7-1-1 | I/O | Input port | TOOL0 | Port 4.<br>4-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>P41 can be set to analog input[Note]. |
| P41 | 7-3-1 | | Analog input port | ANI30/8TI07/TO07 | |
| P42 | 7-1-1 | | Input port | TI04/TO04 | |
| P43 | | | | – | |
| P50 | 7-3-2 | I/O | Analog input port | ANI26/INTP1/SI11/SDA11 | Port 5.<br>2-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Output of P50 can be set to N-ch open-drain output ($V_{DD}$ tolerance).<br>Can be set to analog input[Note]. |
| P51 | 7-3-1 | | | ANI25/INTP2/SO11 | |
| P60 | 12-1-1 | I/O | Input port | SCLA0 | Port 6.<br>4-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>N-ch open-drain output (6 V tolerance). |
| P61 | | | | SDAA0 | |
| P62 | | | | – | |
| P63 | | | | – | |
| P70 | 7-3-1 | I/O | Analog input port | ANI28/KR0/SCK21/SCL21 | Port 7.<br>8-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>Output of P71 and P74 can be set to N-ch open-drain output ($V_{DD}$ tolerance).<br>P70 can be set to analog input[Note]. |
| P71 | 7-1-2 | | Input port | KR1/SI21/SDA21 | |
| P72 | 7-1-1 | | | KR2/SO21 | |
| P73 | | | | KR3/SO01 | |
| P74 | 7-1-2 | | | KR4/INTP8/SI01/SDA01 | |
| P75 | 7-1-1 | | | KR5/INTP9/SCK01/SCL01 | |
| P76 | | | | KR6/INTP10 | |
| P77 | | | | KR7/INTP11 | |
| P120 | 7-3-1 | I/O | Analog input port | ANI19 | Port 12.<br>1-bit I/O port and 4-bit input only port.<br>For only P120, input/output can be specified.<br>For only P120, use of an on-chip pull-up resistor can be specified by a software setting at input port.<br>P120 can be set to analog input[Note]. |
| P121 | 2-2-1 | Input | Input port | X1 | |
| P122 | | | | X2/EXCLK | |
| P123 | | | | XT1 | |
| P124 | | | | XT2/EXCLKS | |
| P130 | 1-1-1 | Output | Output port | – | Port 13.<br>1-bit output port and 1-bit input only port. |
| P137 | 2-1-2 | Input | Input port | INTP0 | |
| P140 | 7-1-1 | I/O | Input port | PCLBUZ0/INTP6 | Port 14.<br>2-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Use of an on-chip pull-up resistor can be specified by a software setting at input port. |
| P141 | | | | PCLBUZ1/INTP7 | |

**Note** Digital or analog for each pin can be selected with the port mode control register x (PMCx) (can be set in 1-bit units).

(3/3)

<R>

| Function Name | Pin Type | I/O | After Reset | Alternate Function | Function |
|---|---|---|---|---|---|
| P150 | 4-3-1 | I/O | Analog input port | ANI8 | Port 15.<br>5-bit I/O port.<br>Input/output can be specified in 1-bit units.<br>Can be set to analog input[Note]. |
| P151 | | | | ANI9/(KR6) | |
| P152 | | | | ANI10/(KR7) | |
| P153 | | | | ANI11/(KR8) | |
| P154 | | | | ANI12/(KR9) | |
| $\overline{\text{RESET}}$ | 2-1-1 | Input | – | – | Input only pin for external reset<br>When external reset is not used, connect this pin to $V_{DD}$ directly or via a resistor. |

**Note** Digital or analog for each pin can be selected with the A/D port configuration register (ADPC).

**Remark** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR). See **Figure 4-8 Format of Peripheral I/O Redirection Register (PIOR)**.

## 2.2 Functions Other than Port Pins

### 2.2.1 With functions for each product

(1/4)

| Function Name | 64-pin | 48-pin | 32-pin | 25-pin |
|---|---|---|---|---|
| ANI0 | √ | √ | √ | √ |
| ANI1 | √ | √ | √ | √ |
| ANI2 | √ | √ | √ | √ |
| ANI3 | √ | √ | √ | √ |
| ANI4 | √ | √ | √ | – |
| ANI5 | √ | √ | – | – |
| ANI6 | √ | √ | – | – |
| ANI7 | √ | √ | – | – |
| ANI8 | √ | √ | – | – |
| ANI9 | √ | – | – | – |
| ANI10 | √ | – | – | – |
| ANI11 | √ | – | – | – |
| ANI12 | √ | – | – | – |
| ANI16 | √ | √ | √ | √ |
| ANI17 | √ | √ | √ | √ |
| ANI18 | √ | √ | √ | √ |
| ANI19 | √ | √ | √ | – |
| ANI20 | √ | √ | √ | √ |
| ANI21 | √ | √ | √ | √ |
| ANI22 | √ | √ | √ | – |
| ANI23 | √ | √ | √ | – |
| ANI24 | √ | √ | √ | – |
| ANI25 | √ | √ | – | √ |
| ANI26 | √ | √ | √ | √ |
| ANI27 | √ | √ | √ | √ |
| ANI28 | √ | √ | √ | – |
| ANI29 | √ | √ | √ | √ |
| ANI30 | √ | √ | – | – |
| INTP0 | √ | √ | √ | √ |
| INTP1 | √ | √ | √ | √ |
| INTP2 | √ | √ | √ | √ |
| INTP3 | √ | √ | √ | √ |
| INTP4 | √ | √ | √ | √ |
| INTP5 | √ | √ | – | – |
| INTP6 | √ | √ | – | – |
| INTP7 | √ | – | – | – |
| INTP8 | √ | √ | – | – |
| INTP9 | √ | √ | – | – |
| INTP10 | √ | – | – | – |
| INTP11 | √ | – | – | – |

(2/4)

| Function Name | 64-pin | 48-pin | 32-pin | 25-pin |
|---|---|---|---|---|
| KR0 | √ | √ | √ | (√) |
| KR1 | √ | √ | (√) | (√) |
| KR2 | √ | √ | (√) | (√) |
| KR3 | √ | √ | (√) | (√) |
| KR4 | √ | √ | (√) | – |
| KR5 | √ | √ | (√) | – |
| KR6 | √ | – | – | – |
| KR7 | √ | – | – | – |
| KR8 | √ | – | – | – |
| KR9 | √ | – | – | – |
| PCLBUZ0 | √ | √ | √ | √ |
| PCLBUZ1 | √ | √ | √ | – |
| REGC | √ | √ | √ | √ |
| RTC1HZ | √ | √ | – | – |
| $\overline{\text{RESET}}$ | √ | √ | √ | √ |
| RxD0 | √ | √ | √ | √ |
| RxD1 | √ | √ | √ | √ |
| RxD2 | √ | √ | √ | – |
| SCK00 | √ | √ | √ | √ |
| SCK01 | √ | √ | – | – |
| SCK10 | √ | – | – | – |
| SCK11 | √ | √ | √ | √ |
| SCK20 | √ | √ | √ | – |
| SCK21 | √ | √ | – | – |
| SCLA0 | √ | √ | √ | √ |
| SCL00 | √ | √ | √ | √ |
| SCL01 | √ | √ | – | – |
| SCL10 | √ | – | – | – |
| SCL11 | √ | √ | √ | √ |
| SCL20 | √ | √ | √ | – |
| SCL21 | √ | √ | – | – |
| SDAA0 | √ | √ | √ | √ |
| SDA00 | √ | √ | √ | √ |
| SDA01 | √ | √ | – | – |
| SDA10 | √ | – | – | – |
| SDA11 | √ | √ | √ | √ |
| SDA20 | √ | √ | √ | – |
| SDA21 | √ | √ | – | – |

**Remark** The checked function is available only when the bit corresponding to the function in the peripheral I/O redirection register (PIOR) is set to 1.

(3/4)

| Function Name | 64-pin | 48-pin | 32-pin | 25-pin |
|---|---|---|---|---|
| SI00 | √ | √ | √ | √ |
| SI01 | √ | √ | – | – |
| SI10 | √ | – | – | – |
| SI11 | √ | √ | √ | √ |
| SI20 | √ | √ | √ | – |
| SI21 | √ | √ | – | – |
| SO00 | √ | √ | √ | √ |
| SO01 | √ | √ | – | – |
| SO10 | √ | – | – | – |
| SO11 | √ | √ | √ | √ |
| SO20 | √ | √ | √ | – |
| SO21 | √ | √ | – | – |
| TI01 | √ | √ | – | – |
| TI03 | √ | √ | √ | √ |
| TI04 | √ | – | – | – |
| TI05 | √ | – | – | – |
| TI06 | √ | – | – | – |
| TI07 | √ | √ | – | – |
| TO00 | √ | √ | √ | √ |
| TO01 | √ | √ | – | – |
| TO03 | √ | √ | √ | √ |
| TO04 | √ | – | – | – |
| TO05 | √ | – | – | – |
| TO06 | √ | – | – | – |
| TO07 | √ | √ | – | – |
| TxD0 | √ | √ | √ | √ |
| TxD1 | √ | √ | √ | √ |
| TxD2 | √ | √ | √ | – |
| X1 | √ | √ | √ | √ |
| X2 | √ | √ | √ | √ |
| EXCLK | √ | √ | √ | √ |
| EXCLKS | √ | √ | – | – |
| TI00 | √ | √ | √ | √ |
| XT1 | √ | √ | – | – |
| XT2 | √ | √ | – | – |

(4/4)

| Function Name | 64-pin | 48-pin | 32-pin | 25-pin |
|---|---|---|---|---|
| $V_{DD}$ | √ | √ | √ | √ |
| $EV_{DD0}$ | √ | – | – | – |
| $AV_{DD}$ | √ | √ | √ | √ |
| $AV_{REFP}$ | √ | √ | √ | √ |
| $AV_{REFM}$ | √ | √ | √ | √ |
| $V_{SS}$ | √ | √ | √ | √ |
| $EV_{SS0}$ | √ | – | – | – |
| $AV_{SS}$ | √ | √ | √ | √ |
| TOOLRxD | √ | √ | √ | √ |
| TOOLTxD | √ | √ | √ | √ |
| TOOL0 | √ | √ | √ | √ |

### 2.2.2 Explanation of function

(1/2)

| Function Name | I/O | Function |
|---|---|---|
| ANI0 to ANI12, ANI16 to ANI30 | Input | A/D converter analog input (see **Figure 11-44 Analog Input Pin Connection**) |
| INTP0 to INTP11 | Input | External interrupt request input<br>The valid edge can be specified : Rising edge, falling edge, or both rising and falling edges) |
| KR0 to KR9 | Input | Key interrupt input |
| PCLBUZ0, PCLBUZ1 | Output | Clock output/buzzer output |
| REGC | – | Pin for connecting regulator output stabilization capacitance for internal operation.<br>Connect this pin to $V_{SS}$ via a capacitor (0.47 to 1 $\mu$F).<br>Also, use a capacitor with good characteristics, since it is used to stabilize internal voltage. |
| RTC1HZ | Output | Real-time clock correction clock (1 Hz) output |
| $\overline{\text{RESET}}$ | Input | This is the active-low system reset input pin.<br>When the external reset is not used, connect this pin directly or via a resistor to $V_{DD}$. |
| RxD0 to RxD2 | Input | Serial data input pins of serial interface UART0, UART1, and UART2 |
| TxD0 to TxD2 | Output | Serial data output pins of serial interface UART0, UART1, and UART2 |
| SCK00, SCK01, SCK10, SCK11, SCK20, SCK21 | I/O | Serial clock I/O pins of serial interface CSI00, CSI01, CSI10, CSI11, CSI20, and CSI21 |
| SCL00, SCL01, SCL10, SCL11, SCL20, SCL21 | Output | Serial clock output pins of serial interface IIC00, IIC01, IIC10, IIC11, IIC20, and IIC21 |
| SDA00, SDA01, SDA10, SDA11, SDA20, SDA21 | I/O | Serial data I/O pins of serial interface IIC00, IIC01, IIC10, IIC11, IIC20, and IIC21 |
| SI00, SI01, SI10, SI11, SI20, SI21 | Input | Serial data input pins of serial interface CSI00, CSI01, CSI10, CSI11, CSI20, and CSI21 |
| SO00, SO01, SO10, SO11, SO20, SO21 | Output | Serial data output pins of serial interface CSI00, CSI01, CSI10, CSI11, CSI20, and CSI21 |
| SCLA0 | I/O | Serial clock I/O pins of serial interface IICA0 |
| SDAA0 | I/O | Serial data I/O pins of serial interface IICA0 |
| <R> TI00, TI01, TI03 to TI07 | Input | The pins for inputting an external count clock/capture trigger to 16-bit timers 00, 01, and 03 to 07 |
| TO00, TO01, TO03 to TO07 | Output | Timer output pins of 16-bit timers 00, 01, and 03 to 07 |
| X1, X2 | – | Resonator connection for main system clock |
| EXCLK | Input | External clock input for main system clock |
| XT1, XT2 | – | Resonator connection for subsystem clock |
| EXCLKS | Input | External clock input for subsystem clock |

(2/2)

| Function Name | I/O | Function |
|---|---|---|
| $V_{DD}$ | – | <25-pin, 32-pin 48-pin><br>Positive power supply for port pin other than P20 to P27, P150 and $\overline{\text{RESET}}$, REGC pin.<br><64-pin ><br>Positive power supply for P121 to P124, P137 and $\overline{\text{RESET}}$, REGC pin. |
| $EV_{DD0}$ | – | Positive power supply for ports (other than P20 to P27, P121 to P124, P137, P150 to P154) |
| $AV_{DD}$ | – | Positive power supply for P20 to P27, P150 to P154, and A/D converter |
| $AV_{REFP}$ | Input | A/D converter reference potential (+ side) input |
| $AV_{REFM}$ | Input | A/D converter reference potential (– side) input<br>Make $AV_{REFM}$ pin the same potential as $AV_{SS}$ and $V_{SS}$ pin. |
| $V_{SS}$ | – | <25-pin, 32-pin, 48-pin><br>Ground potential for port pin other than P20 to P27, P150 and $\overline{\text{RESET}}$, REGC pin.<br><64-pin><br>Ground potential for P121 to P124, P137 and $\overline{\text{RESET}}$, REGC pin. |
| $EV_{SS0}$ | – | Ground potential for ports (other than P20 to P27, P121 to P124, P137, P150 to P154)<br>Make $EV_{SS0}$ pin the same potential as $V_{SS}$ pin. |
| $AV_{SS}$ | – | Ground potential for A/D converter, P20 to P27, and P150 to P154. Use this pin with the same potential as $EV_{SS0}$, and $V_{SS}$. |
| TOOLRxD | Input | UART reception pin for the external device connection used during flash memory programming |
| TOOLTxD | Output | UART transmission pin for the external device connection used during flash memory programming |
| TOOL0 | I/O | Data I/O for flash memory programmer/debugger |

**Caution After reset release, the relationships between P40/TOOL0 and the operating mode are as follows.**

**Table 2-2. Relationships Between P40/TOOL0 and Operation Mode After Reset Release**

| P40/TOOL0 | Operating Mode |
|---|---|
| $EV_{DD0}$ | Normal operation mode |
| 0 V | Flash memory programming mode |

**For details, see 25.4 Serial Programming Method.**

**Remark** Use bypass capacitors (about 0.1 $\mu$F) as noise and latch up countermeasures with relatively thick wires at the shortest distance to $V_{DD}$ to $V_{SS}$, $EV_{DD0}$ to $EV_{SS0}$ lines.

## <R> 2.3 Pin I/O Circuits and Recommended Connection of Unused Pins

Table 2-3 shows the connections of unused pins.

**Remark** The pins mounted depend on the product. See **1.3 Pin Configuration (Top View)** and **2.1 Port Function.**

<R> **Table 2-3. Connection of Unused Pins**

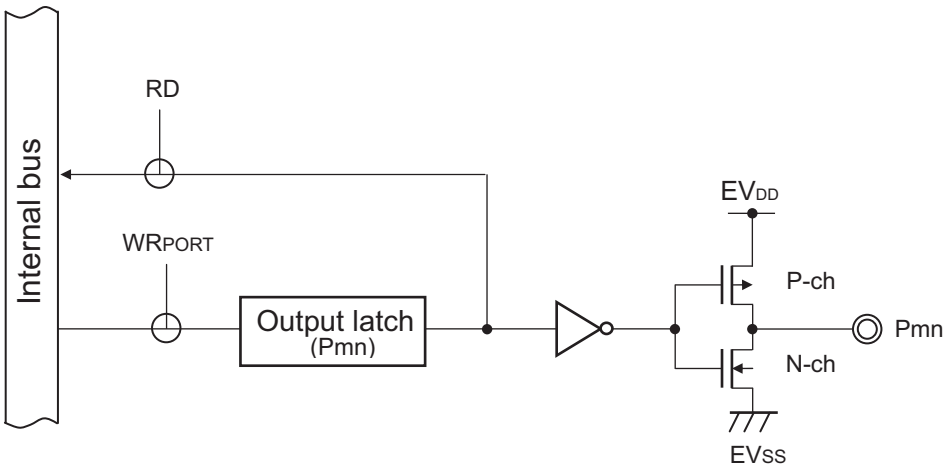
| Pin Name | I/O | Recommended Connection of Unused Pins |
|---|---|---|
| P00 to P06 | I/O | Input: Independently connect to $EV_{DD0}$ or $EV_{SS0}$ via a resistor.<br>Output: Leave open. |
| P10 to P16 | | |
| P20 to P27 | | Input: Independently connect to $AV_{DD}$ or $AV_{SS}$ via a resistor.<br>Output: Leave open. |
| P30, P31 | | Input: Independently connect to $EV_{DD0}$ or $EV_{SS0}$ via a resistor.<br>Output: Leave open. |
| P40/TOOL0 | | Input: Independently connect to $EV_{DD0}$ or leave open.<br>Output: Leave open. |
| P41 to P43 | | Input: Independently connect to $EV_{DD0}$ or $EV_{SS0}$ via a resistor.<br>Output: Leave open. |
| P50, P51 | | |
| P60 to P63 | | Input: Independently connect to $EV_{DD0}$ or $EV_{SS0}$ via a resistor.<br>Output: Set the port's output latch to 0 and leave the pins open, or set the port's output latch to 1 and independently connect the pins to $EV_{DD0}$ or $EV_{SS0}$ via a resistor. |
| P70 to P77 | | Input: Independently connect to $EV_{DD0}$ or $EV_{SS0}$ via a resistor.<br>Output: Leave open. |
| P120 | | Input: Independently connect to $EV_{DD0}$ or $EV_{SS0}$ via a resistor.<br>Output: Leave open. |
| P121 to P124 | Input | Independently connect to $V_{DD}$ or $V_{SS}$ via a resistor. |
| P130 | Output | Leave open. |
| P137 | Input | Independently connect to $V_{DD}$ or $V_{SS}$ via a resistor. |
| P140, P141 | I/O | Input: Independently connect to $EV_{DD0}$ or $EV_{SS0}$ via a resistor.<br>Output: Leave open. |
| P150 to P154 | | Input: Independently connect to $AV_{DD}$ or $AV_{SS}$ via a resistor.<br>Output: Leave open. |
| $\overline{\text{RESET}}$ | Input | Connect directly or via a resistor to $V_{DD}$. |
| REGC | – | Connect to $V_{SS}$ via capacitor (0.47 to 1 $\mu$F). |

**Remark** With products not provided with an $EV_{DD0}$, or $EV_{SS0}$ pin, replace $EV_{DD0}$ with $V_{DD}$, or replace $EV_{SS0}$ with $V_{SS}$.

## <R> 2.4 Block Diagrams of Pins

Figures 2-1 to 2-13 show the block diagrams of the pins described in **2.1.1 25-pin products** to **2.1.4 64-pin products**.

<R> **Figure 2-1. Pin Block Diagram for Pin Type 1-1-1**




<R> **Figure 2-2. Pin Block Diagram for Pin Type 2-1-1**

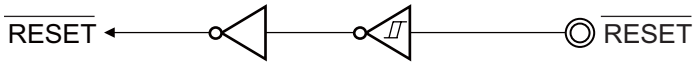
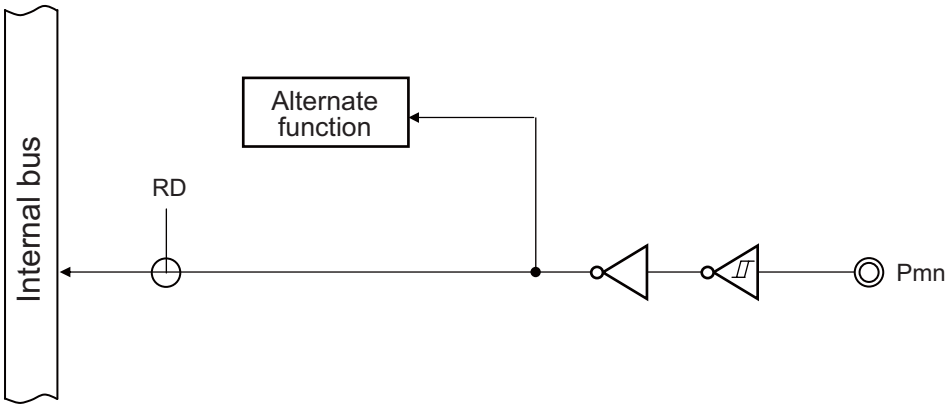



**Figure 2-3. Pin Block Diagram for Pin Type 2-1-2**

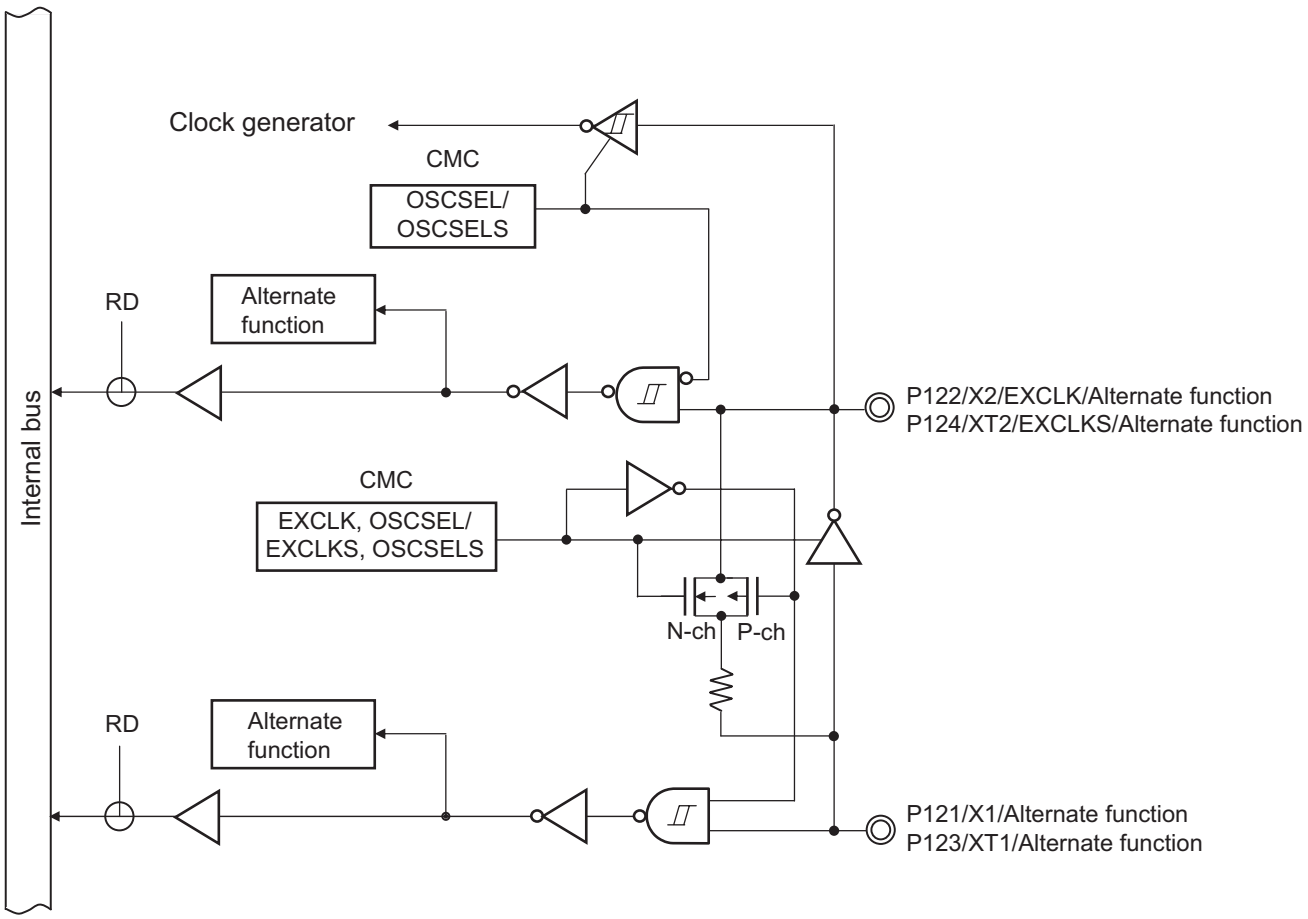
<R>




**Remark** For alternate functions, see **2.1 Port Function**.

<R>

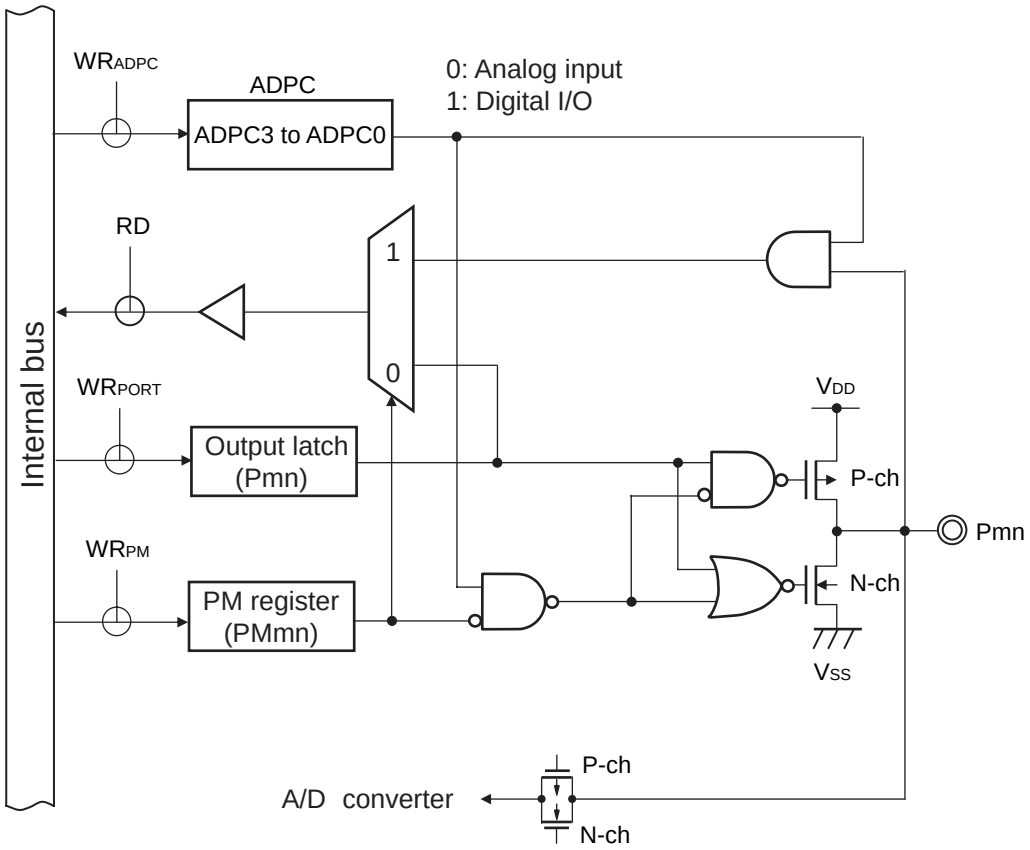
**Figure 2-4. Pin Block Diagram for Pin Type 2-2-1**




**Remark** For alternate functions, see **2.1 Port Function**.

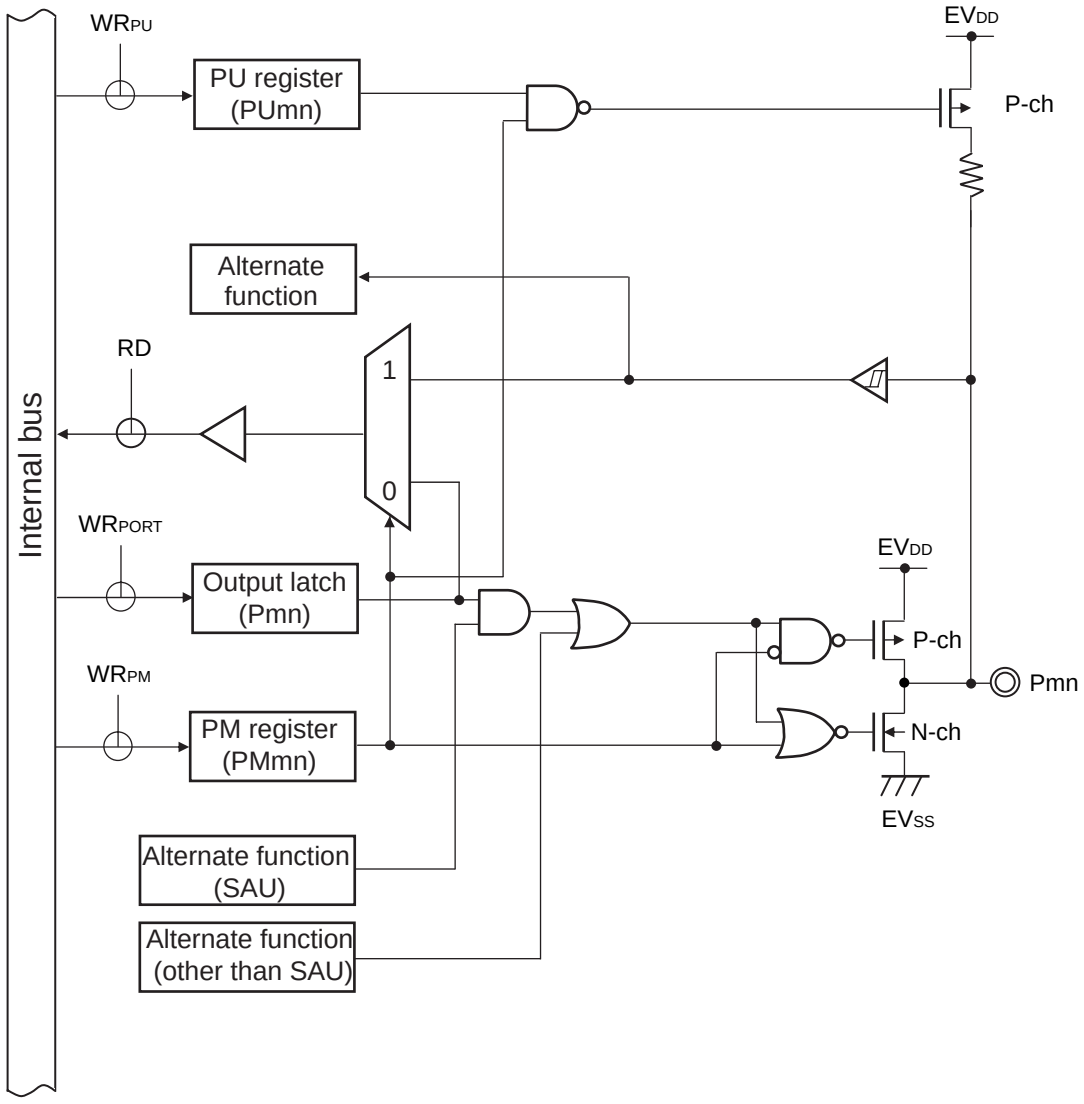
<R>

**Figure 2-5. Pin Block Diagram for Pin Type 4-3-1**



<R>

**Figure 2-6. Pin Block Diagram for Pin Type 7-1-1**

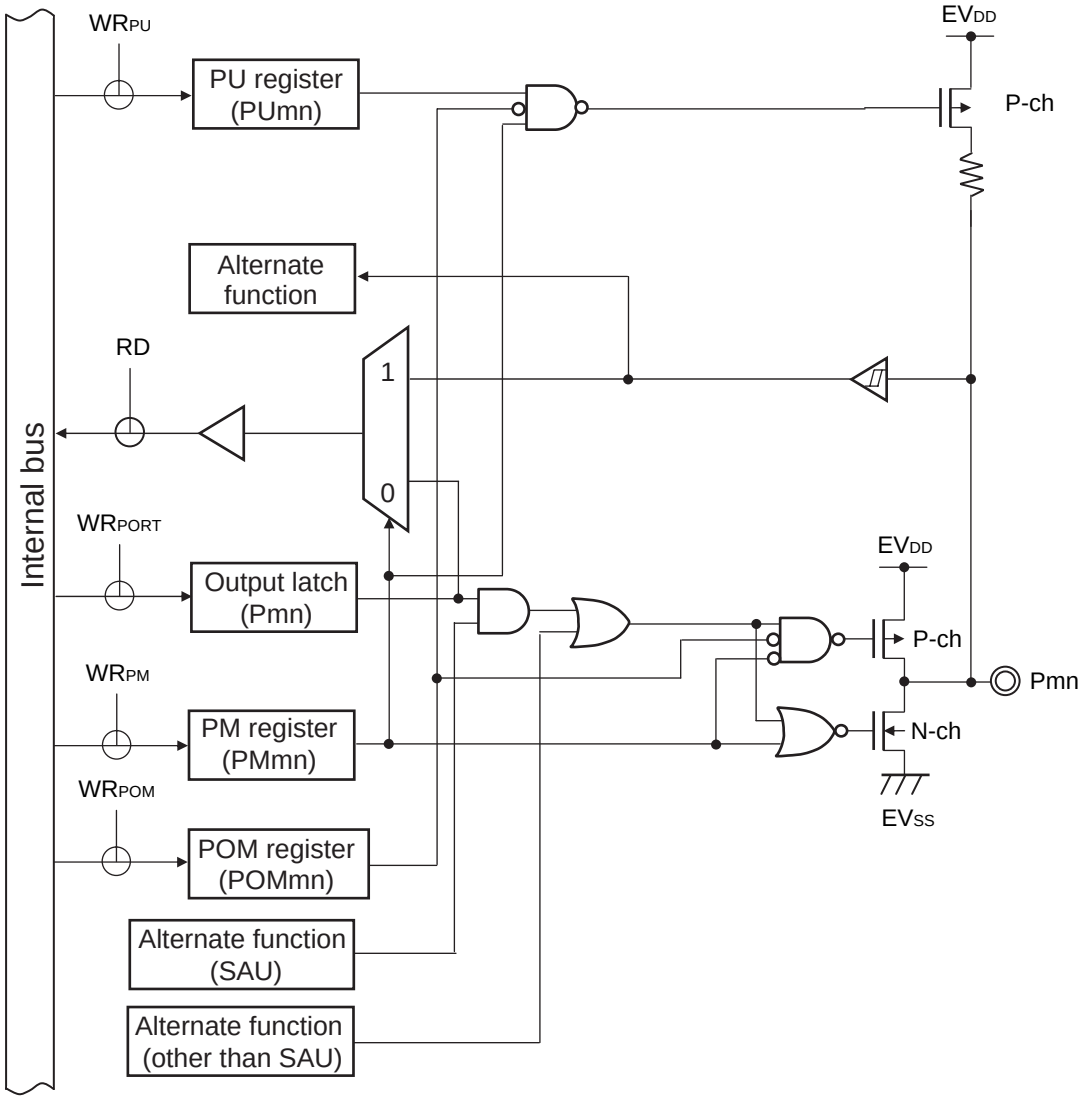



**Remarks 1.** For alternate functions, see **2.1 Port Function**.

**2.** SAU: Serial array unit

<R>

**Figure 2-7. Pin Block Diagram for Pin Type 7-1-2**

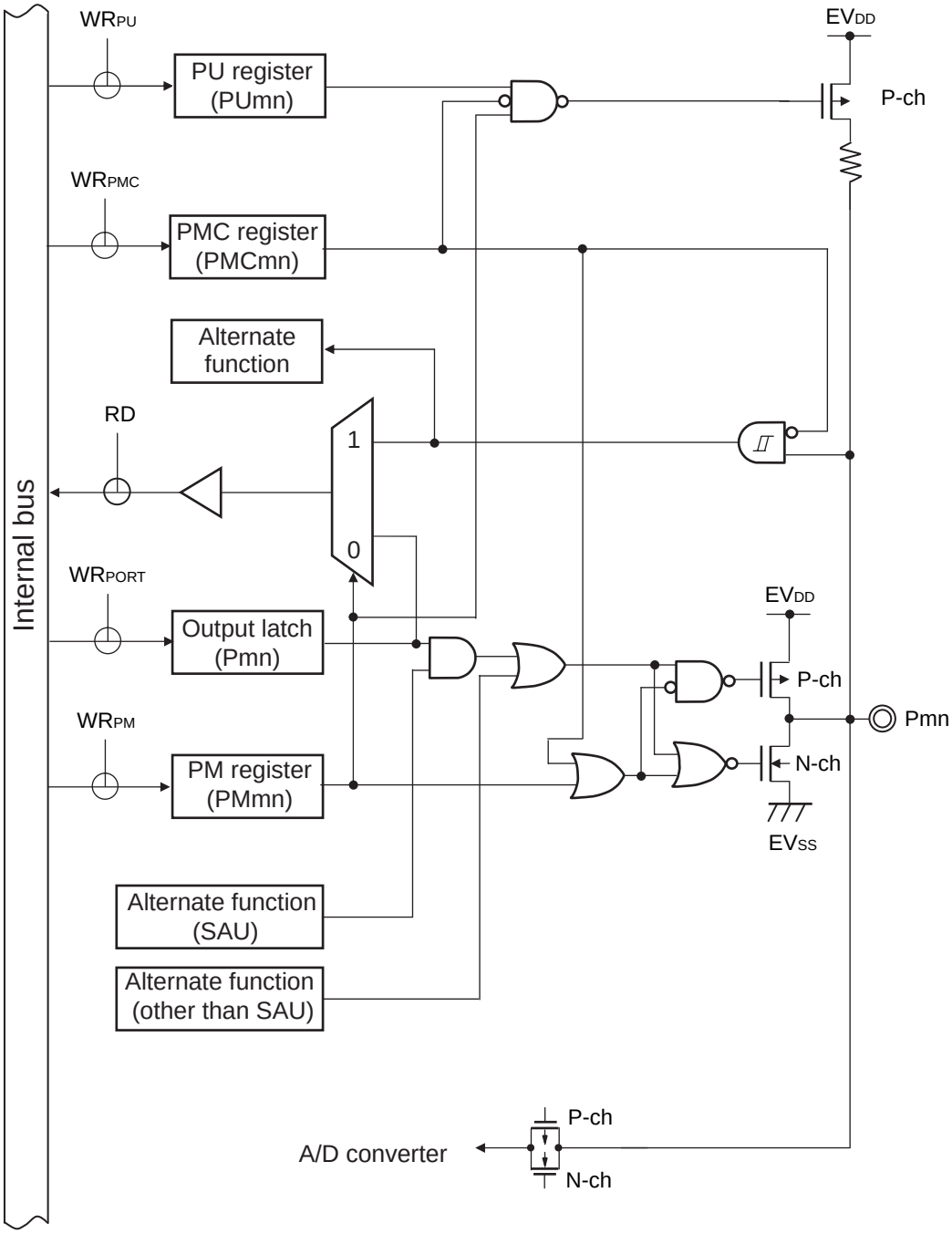



**Remarks 1.** For alternate functions, see **2.1 Port Function**.

**2.** SAU: Serial array unit

<R>

**Figure 2-8. Pin Block Diagram for Pin Type 7-3-1**

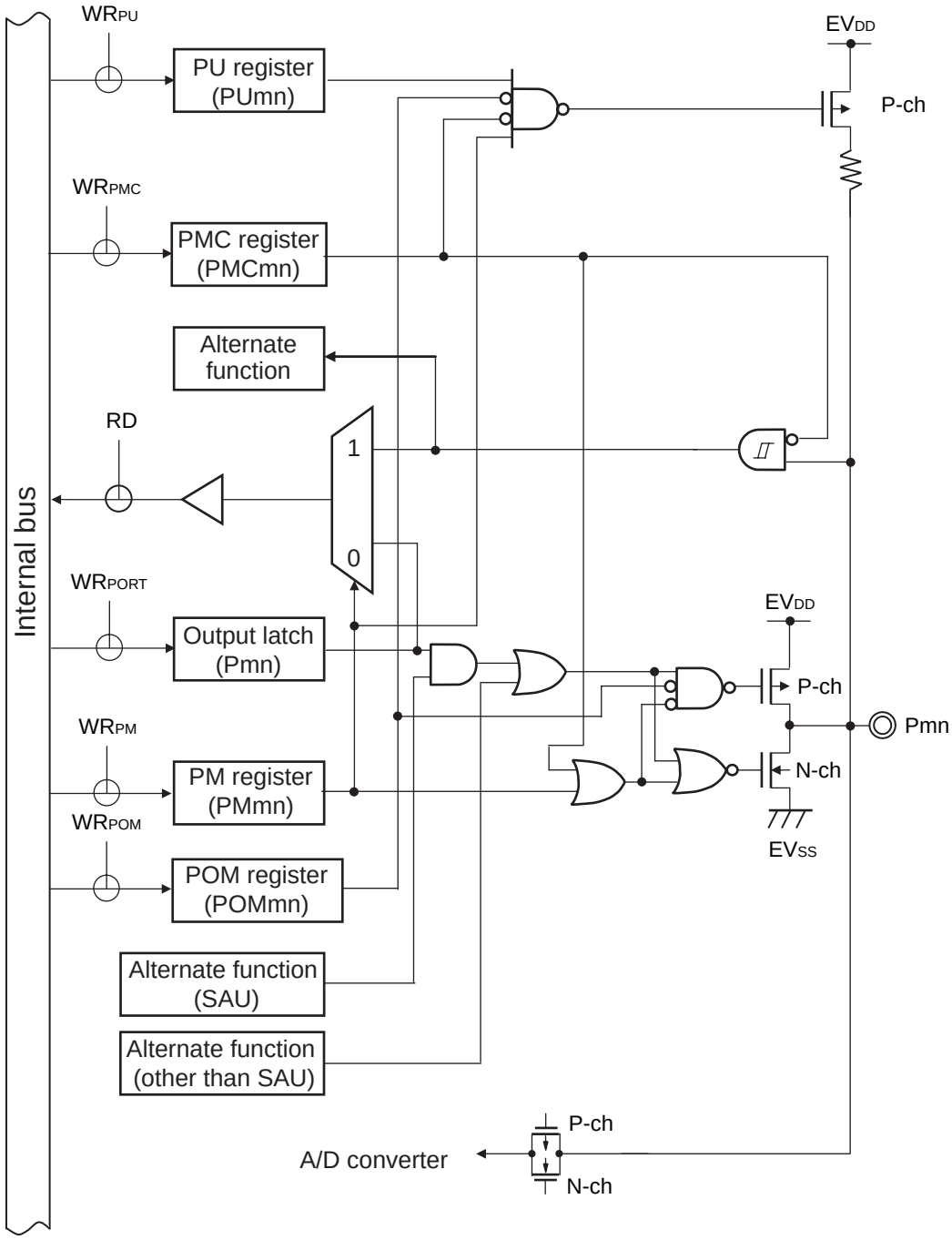



**Remarks 1.** For alternate functions, see **2.1 Port Function**.

**2.** SAU: Serial array unit

<R>

**Figure 2-9. Pin Block Diagram for Pin Type 7-3-2**

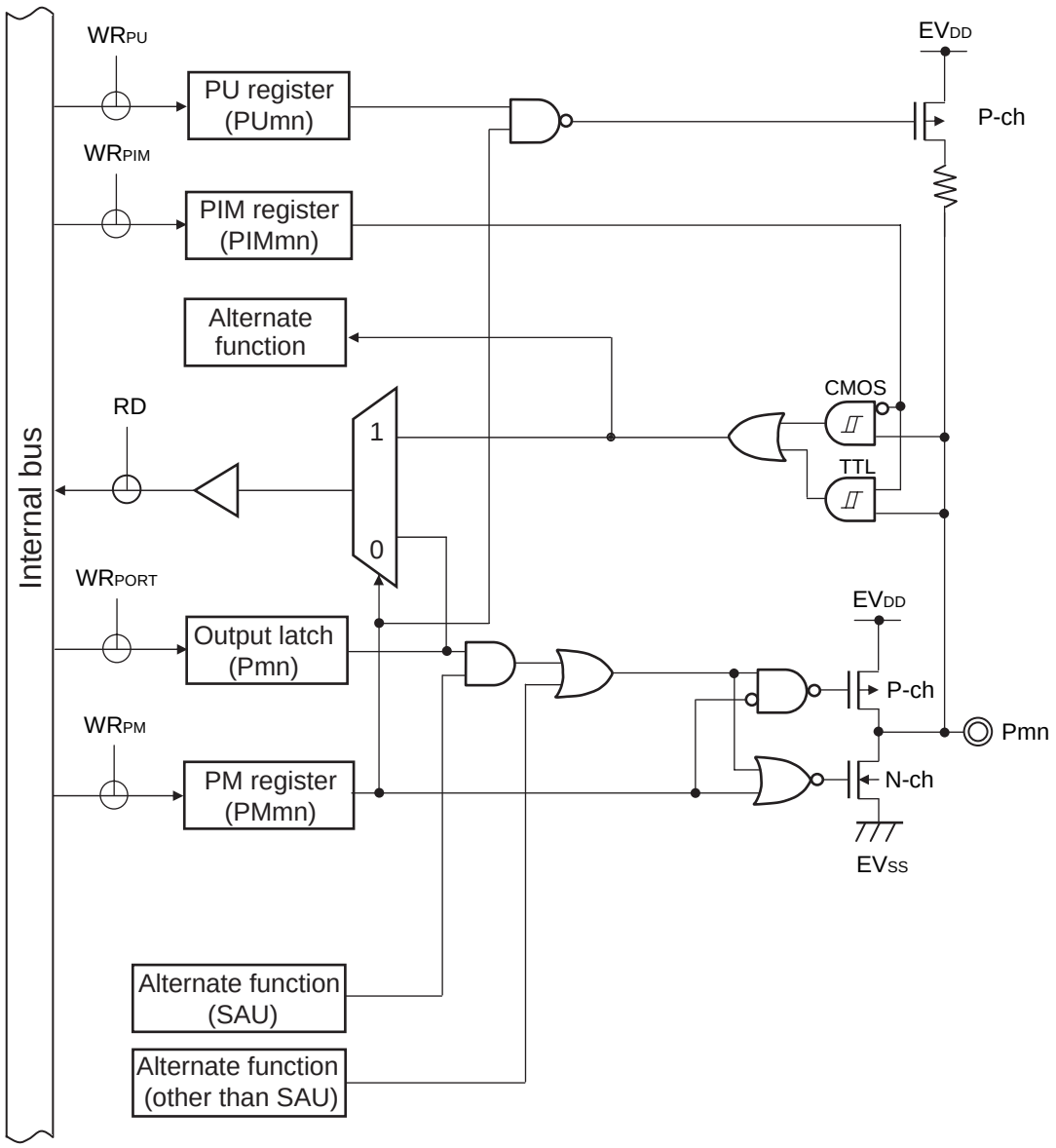



**Remarks 1.** For alternate functions, see **2.1 Port Function**.

**2.** SAU: Serial array unit

<R>

**Figure 2-10. Pin Block Diagram for Pin Type 8-1-1**

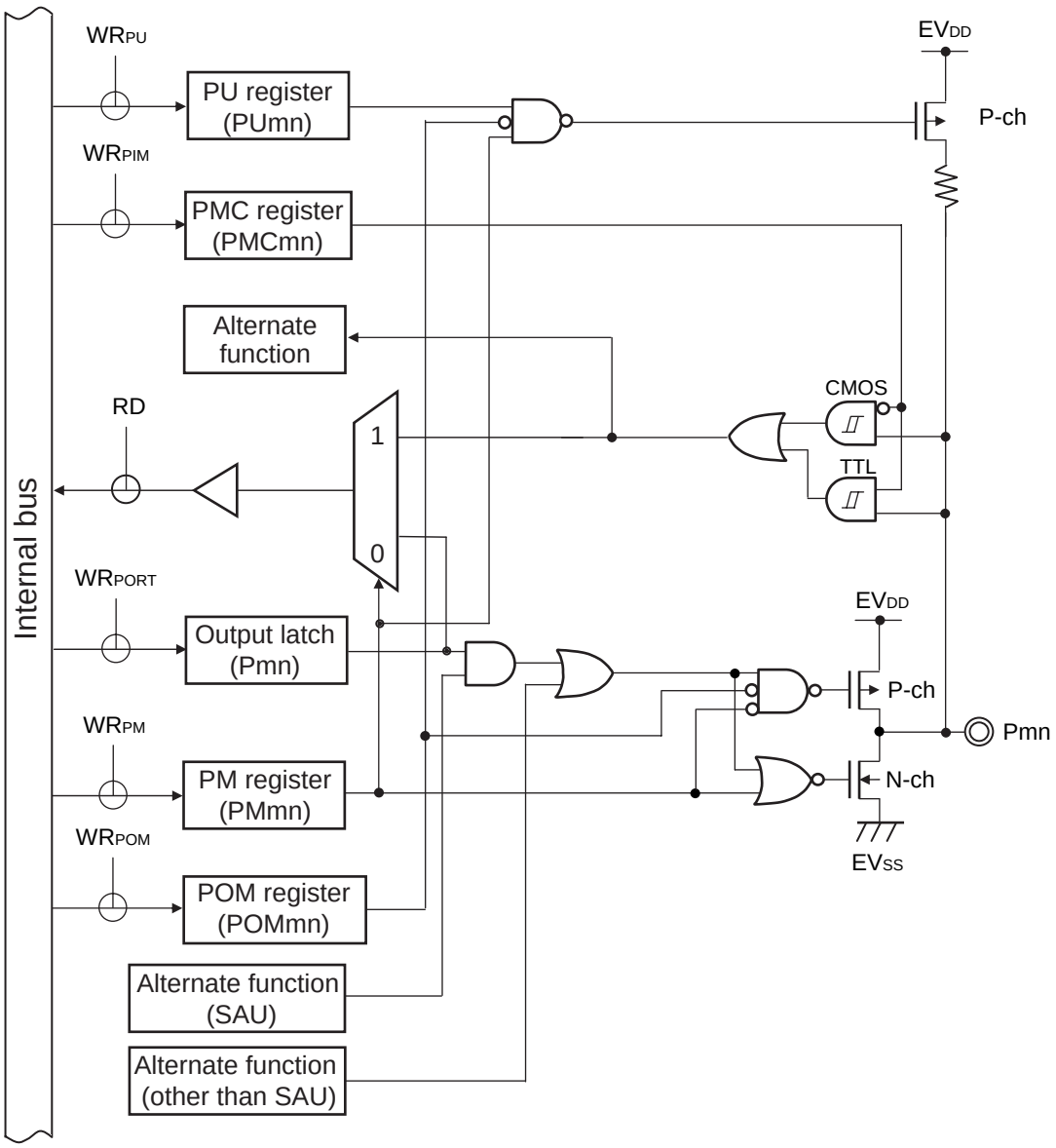



**Remarks 1.** For alternate functions, see **2.1 Port Function**.

**2.** SAU: Serial array unit

<R>

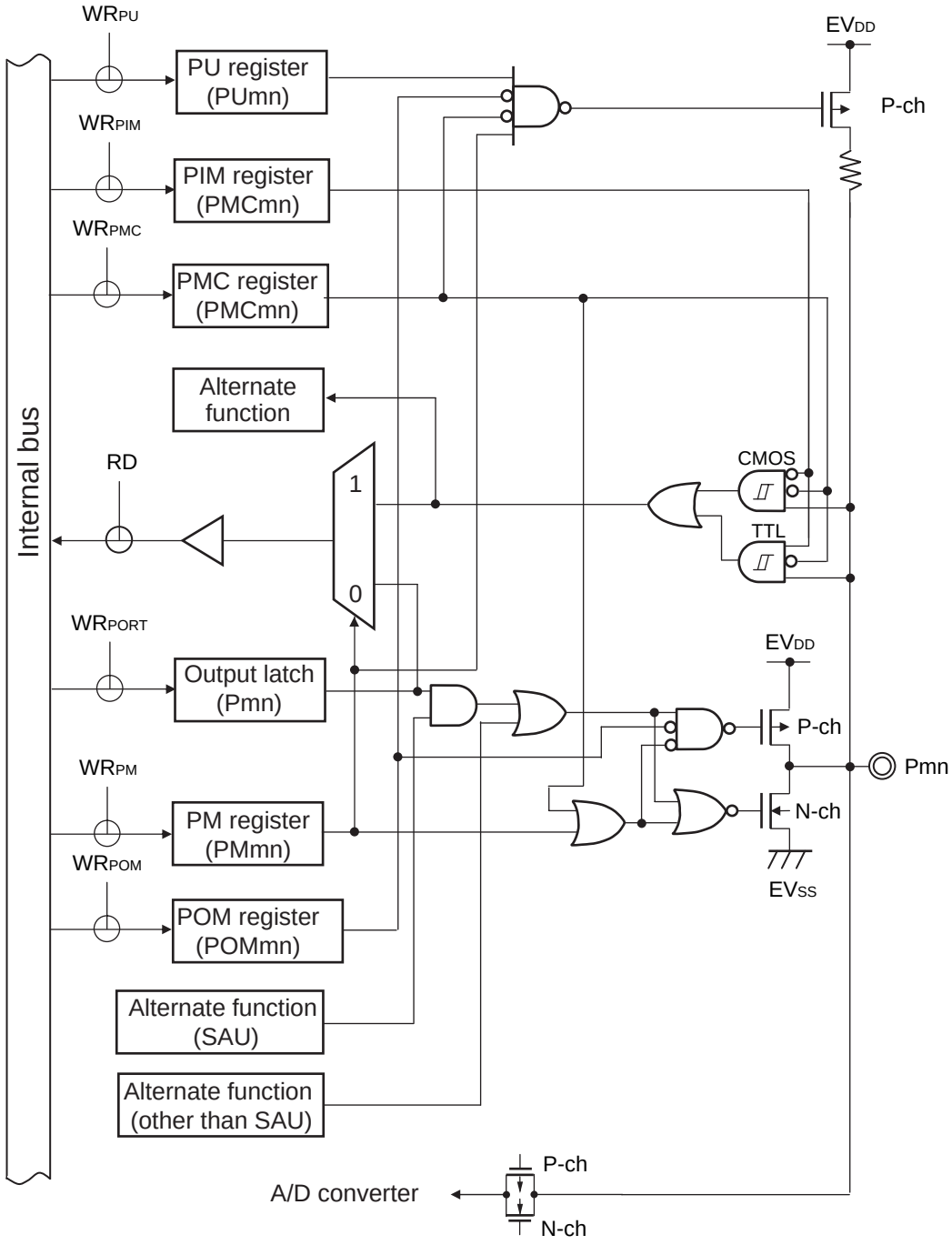
**Figure 2-11. Pin Block Diagram for Pin Type 8-1-2**




**Remarks 1.** For alternate functions, see **2.1 Port Function**.

**2.** SAU: Serial array unit

<R> **Figure 2-12. Pin Block Diagram for Pin Type 8-3-2**

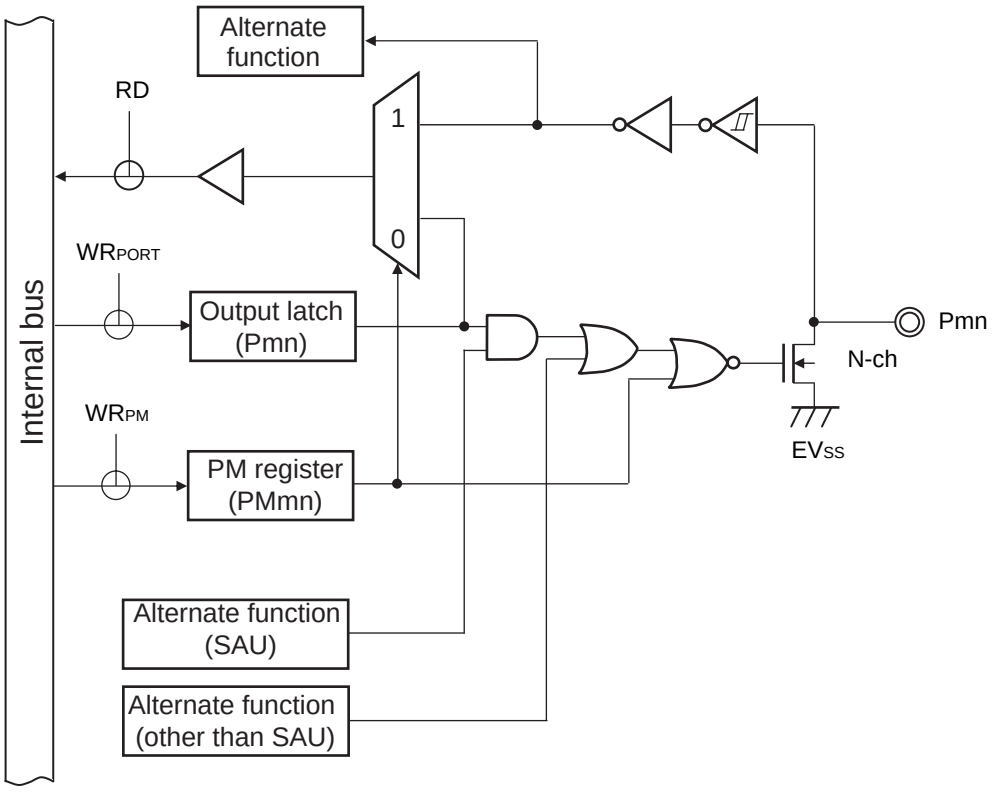



**Remarks 1.** For alternate functions, see **2.1 Port Function**.

**2.** SAU: Serial array unit

<R>

**Figure 2-13. Pin Block Diagram for Pin Type 12-1-1**




**Remarks 1.** For alternate functions, see **2.1 Port Function**.

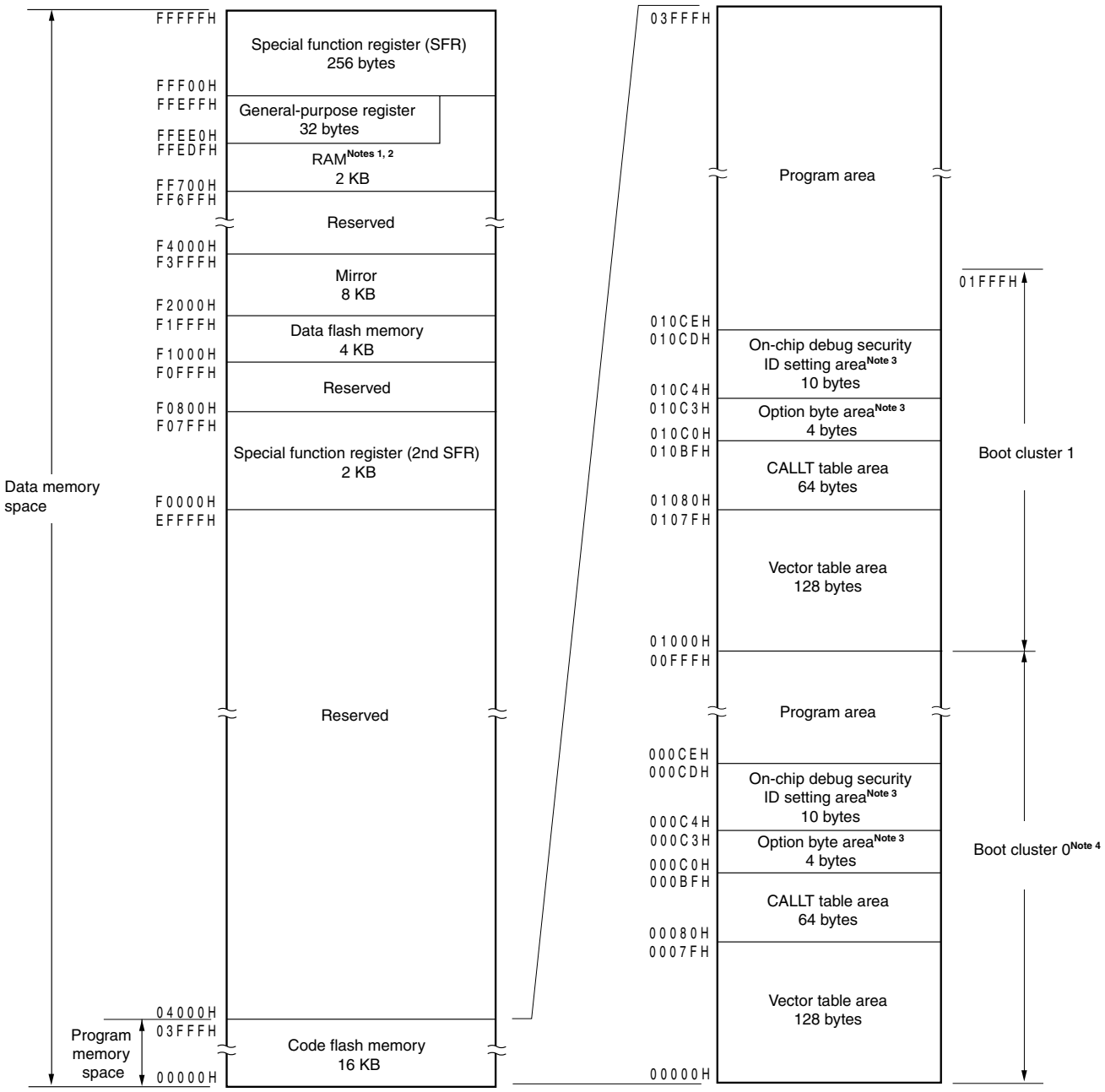
**2.** SAU: Serial array unit

# CHAPTER 3 CPU ARCHITECTURE

## 3.1 Memory Space

Products in the RL78/G1A can access a 1 MB address space. Figures 3-1 to 3-4 show the memory maps.

**Figure 3-1. Memory Map (R5F10ExA (x = 8, B, G))**




**Notes 1.** During self programming and data flash rewriting, the stack, data buffer, and RAM addresses used as branch destinations for vectored interrupts or as sources or destinations of DMA transfers must not be allocated to the area between addresses FFE20H and FFEDFH.

**2.** Instructions can be executed from the RAM area excluding the general-purpose register area.

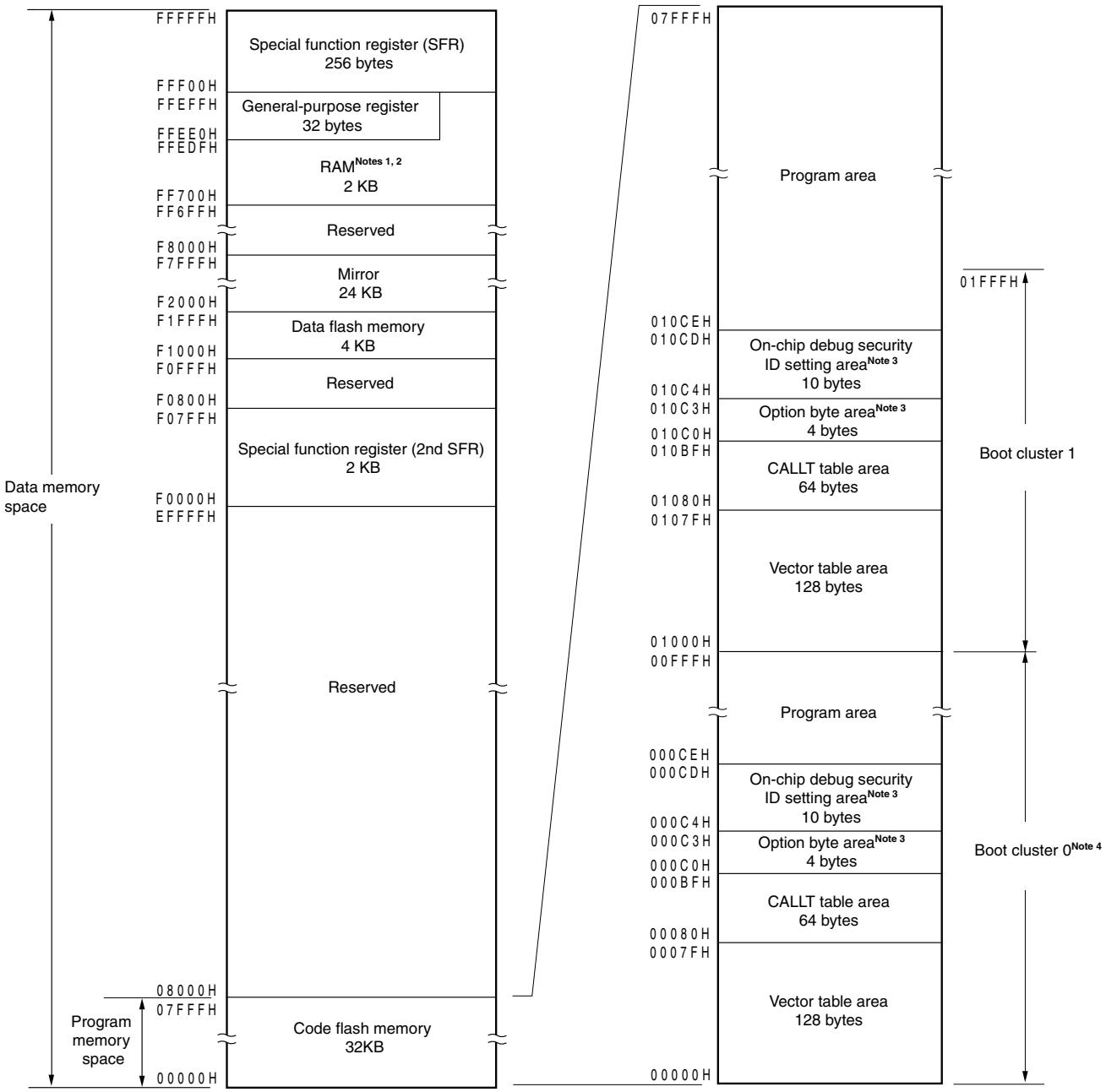
**3.** When boot swap is not used: Set the option bytes to 000C0H to 000C3H, and the on-chip debug security IDs to 000C4H to 000CDH.

When boot swap is used: Set the option bytes to 000C0H to 000C3H and 010C0H to 010C3H, and the on-chip debug security IDs to 000C4H to 000CDH and 010C4H to 010CDH.

**4.** Writing boot cluster 0 can be prohibited depending on the setting of security (see **25.7 Security Setting**).

**Caution While RAM parity error resets are enabled (RPERDIS = 0), be sure to initialize RAM areas where data access is to proceed and the RAM area + 10 bytes when instructions are fetched from RAM areas, respectively. Reset signal generation sets RAM parity error resets to enabled (RPERDIS = 0). For details, see 22.3.3 RAM parity error detection function.**

**Figure 3-2. Memory Map (R5F10ExC (x = 8, B, G, L))**




**Notes 1.** During self programming and data flash rewriting, the stack, data buffer, and RAM addresses used as branch destinations for vectored interrupts or as sources or destinations of DMA transfers must not be allocated to the area between addresses FFE20H and FFEDFH.

**2.** Instructions can be executed from the RAM area excluding the general-purpose register area.

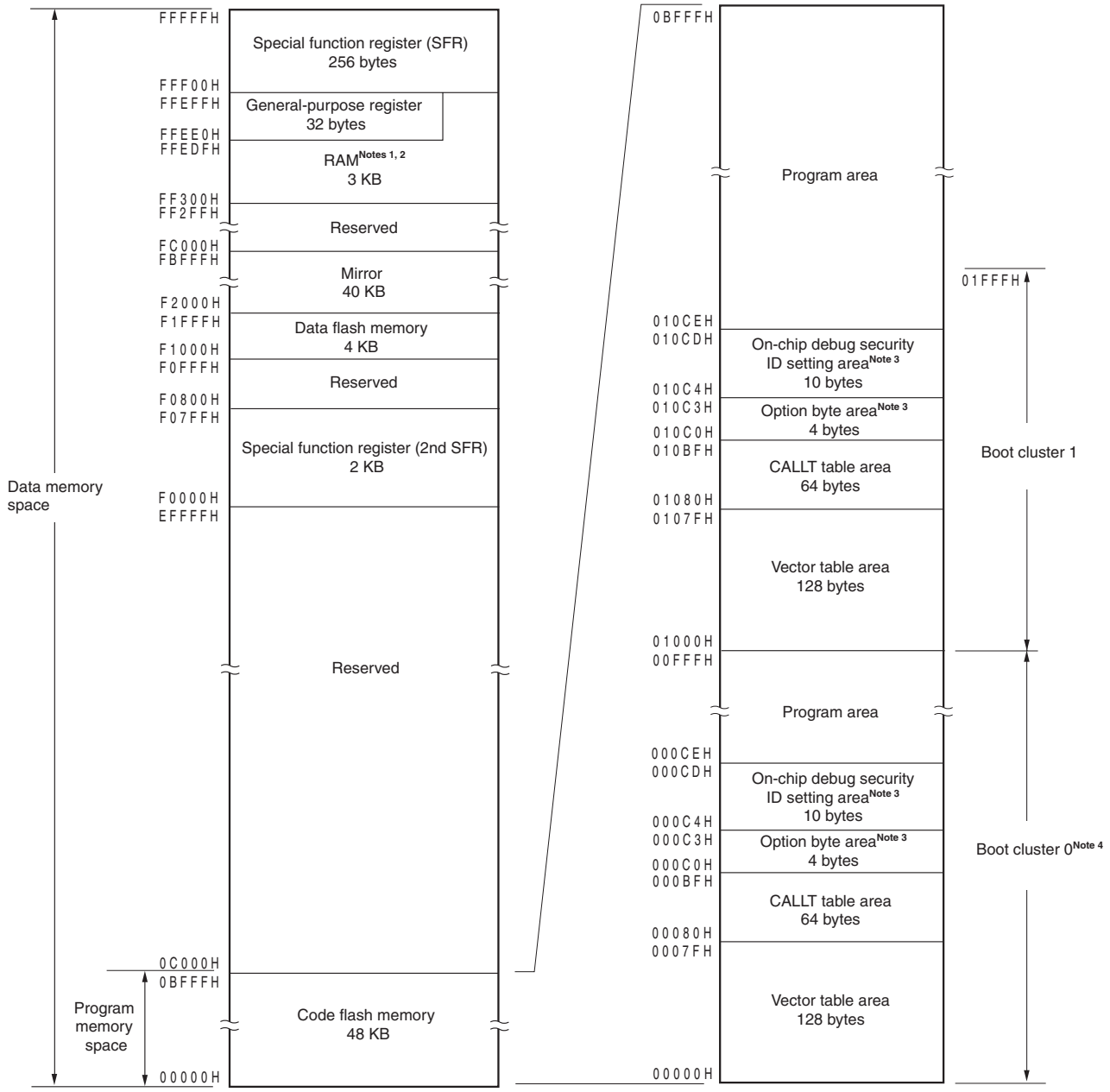
**3.** When boot swap is not used: Set the option bytes to 000C0H to 000C3H, and the on-chip debug security IDs to 000C4H to 000CDH.

When boot swap is used: Set the option bytes to 000C0H to 000C3H and 010C0H to 010C3H, and the on-chip debug security IDs to 000C4H to 000CDH and 010C4H to 010CDH.

**4.** Writing boot cluster 0 can be prohibited depending on the setting of security (see **25.7 Security Setting**).

**Caution While RAM parity error resets are enabled (RPERDIS = 0), be sure to initialize RAM areas where data access is to proceed and the RAM area + 10 bytes when instructions are fetched from RAM areas, respectively. Reset signal generation sets RAM parity error resets to enabled (RPERDIS = 0). For details, see 22.3.3 RAM parity error detection function.**

**Figure 3-3. Memory Map (R5F10ExD (x = 8, B, G, L))**




**Notes 1.** During self programming and data flash rewriting, the stack, data buffer, and RAM addresses used as branch destinations for vectored interrupts or as sources or destinations of DMA transfers must not be allocated to the area between addresses FFE20H and FFEDFH. Also, use of the area FF300H to FF309H is prohibited, because this area is used for each library.

**2.** Instructions can be executed from the RAM area excluding the general-purpose register area.

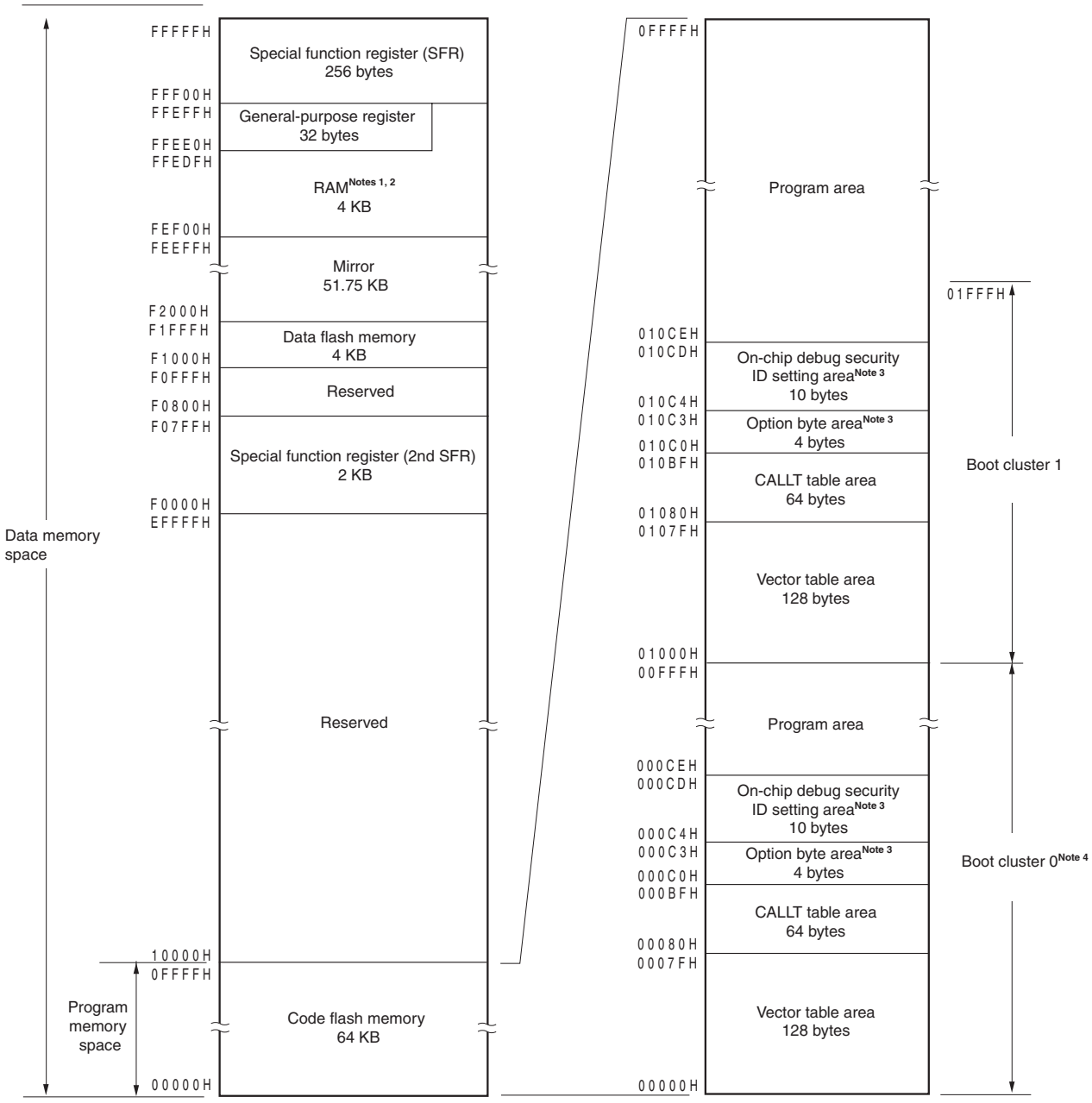
**3.** When boot swap is not used: Set the option bytes to 000C0H to 000C3H, and the on-chip debug security IDs to 000C4H to 000CDH.

When boot swap is used: Set the option bytes to 000C0H to 000C3H and 010C0H to 010C3H, and the on-chip debug security IDs to 000C4H to 000CDH and 010C4H to 010CDH.

**4.** Writing boot cluster 0 can be prohibited depending on the setting of security (see **25.7 Security Setting**).

**Caution While RAM parity error resets are enabled (RPERDIS = 0), be sure to initialize RAM areas where data access is to proceed and the RAM area + 10 bytes when instructions are fetched from RAM areas, respectively. Reset signal generation sets RAM parity error resets to enabled (RPERDIS = 0). For details, see 22.3.3 RAM parity error detection function.**

**Figure 3-4. Memory Map (R5F10ExE (x = 8, B, G, L))**




**Notes 1.** During self programming and data flash rewriting, the stack, data buffer, and RAM addresses used as branch destinations for vectored interrupts or as sources or destinations of DMA transfers must not be allocated to the area between addresses FFE20H and FFEDFH. Also, use of the area FEF00H to FF309H is prohibited, because this area is used for each library.

**2.** Instructions can be executed from the RAM area excluding the general-purpose register area.

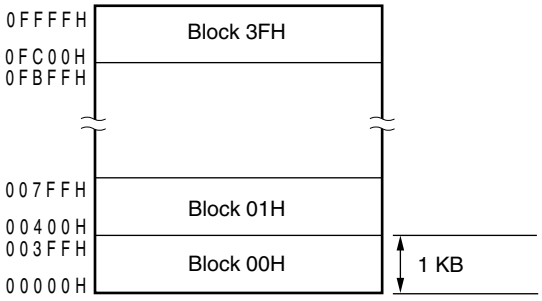
**3.** When boot swap is not used: Set the option bytes to 000C0H to 000C3H, and the on-chip debug security IDs to 000C4H to 000CDH.

When boot swap is used: Set the option bytes to 000C0H to 000C3H and 010C0H to 010C3H, and the on-chip debug security IDs to 000C4H to 000CDH and 010C4H to 010CDH.

**4.** Writing boot cluster 0 can be prohibited depending on the setting of security (see **25.7 Security Setting**).

**Caution While RAM parity error resets are enabled (RPERDIS = 0), be sure to initialize RAM areas where data access is to proceed and the RAM area + 10 bytes when instructions are fetched from RAM areas, respectively. Reset signal generation sets RAM parity error resets to enabled (RPERDIS = 0). For details, see 22.3.3 RAM parity error detection function.**

**Remark The flash memory is divided into blocks (one block = 1 KB). For the address values and block numbers, see Table 3-1 Correspondence Between Address Values and Block Numbers in Flash Memory.**




(R5F10ExE (x = 8, B, G, L))

Correspondence between the address values and block numbers in the flash memory are shown below.

**Table 3-1. Correspondence Between Address Values and Block Numbers in Flash Memory**

| Address Value | Block Number | Address Value | Block Number |
|---|---|---|---|
| 00000H to 003FFH | 00H | 08000H to 083FFH | 20H |
| 00400H to 007FFH | 01H | 08400H to 087FFH | 21H |
| 00800H to 00BFFH | 02H | 08800H to 08BFFH | 22H |
| 00C00H to 00FFFH | 03H | 08C00H to 08FFFH | 23H |
| 01000H to 013FFH | 04H | 09000H to 093FFH | 24H |
| 01400H to 017FFH | 05H | 09400H to 097FFH | 25H |
| 01800H to 01BFFH | 06H | 09800H to 09BFFH | 26H |
| 01C00H to 01FFFH | 07H | 09C00H to 09FFFH | 27H |
| 02000H to 023FFH | 08H | 0A000H to 0A3FFH | 28H |
| 02400H to 027FFH | 09H | 0A400H to 0A7FFH | 29H |
| 02800H to 02BFFH | 0AH | 0A800H to 0ABFFH | 2AH |
| 02C00H to 02FFFH | 0BH | 0AC00H to 0AFFFH | 2BH |
| 03000H to 033FFH | 0CH | 0B000H to 0B3FFH | 2CH |
| 03400H to 037FFH | 0DH | 0B400H to 0B7FFH | 2DH |
| 03800H to 03BFFH | 0EH | 0B800H to 0BBFFH | 2EH |
| 03C00H to 03FFFH | 0FH | 0BC00H to 0BFFFH | 2FH |
| 04000H to 043FFH | 10H | 0C000H to 0C3FFH | 30H |
| 04400H to 047FFH | 11H | 0C400H to 0C7FFH | 31H |
| 04800H to 04BFFH | 12H | 0C800H to 0CBFFH | 32H |
| 04C00H to 04FFFH | 13H | 0CC00H to 0CFFFH | 33H |
| 05000H to 053FFH | 14H | 0D000H to 0D3FFH | 34H |
| 05400H to 057FFH | 15H | 0D400H to 0D7FFH | 35H |
| 05800H to 05BFFH | 16H | 0D800H to 0DBFFH | 36H |
| 05C00H to 05FFFH | 17H | 0DC00H to 0DFFFH | 37H |
| 06000H to 063FFH | 18H | 0E000H to 0E3FFH | 38H |
| 06400H to 067FFH | 19H | 0E400H to 0E7FFH | 39H |
| 06800H to 06BFFH | 1AH | 0E800H to 0EBFFH | 3AH |
| 06C00H to 06FFFH | 1BH | 0EC00H to 0EFFFH | 3BH |
| 07000H to 073FFH | 1CH | 0F000H to 0F3FFH | 3CH |
| 07400H to 077FFH | 1DH | 0F400H to 0F7FFH | 3DH |
| 07800H to 07BFFH | 1EH | 0F800H to 0FBFFH | 3EH |
| 07C00H to 07FFFH | 1FH | 0FC00H to 0FFFFH | 3FH |

**Remark** R5F10ExA (x = 8, B, G) : Block numbers 00H to 0FH
R5F10ExC (x = 8, B, G, L) : Block numbers 00H to 1FH
R5F10ExD (x = 8, B, G, L) : Block numbers 00H to 2FH
R5F10ExE (x = 8, B, G, L) : Block numbers 00H to 3FH

### 3.1.1 Internal program memory space

The internal program memory space stores the program and table data. The RL78/G1A products incorporate internal ROM (flash memory), as shown below.

**Table 3-2. Internal ROM Capacity**

| Part Number | Internal ROM | |
|---|---|---|
| | Structure | Capacity |
| R5F10ExA (x = 8, B, G) | Flash memory | 16384 × 8 bits (00000H to 03FFFH) |
| R5F10ExC (x = 8, B, G, L) | | 32768 × 8 bits (00000H to 07FFFH) |
| R5F10ExD (x = 8, B, G, L) | | 49152 × 8 bits (00000H to 0BFFFH) |
| R5F10ExE (x = 8, B, G, L) | | 65536 × 8 bits (00000H to 0FFFFH) |

The internal program memory space is divided into the following areas.

**(1) Vector table area**

The 128-byte area 00000H to 0007FH is reserved as a vector table area. The program start addresses for branch upon reset or generation of each interrupt request are stored in the vector table area. Furthermore, the interrupt jump address is a 64 K address of 00000H to 0FFFFH, because the vector code is assumed to be 2 bytes.

Of the 16-bit address, the lower 8 bits are stored at even addresses and the higher 8 bits are stored at odd addresses.

To use the boot swap function, set a vector table also at 01000H to 0107FH.

**Table 3-3. Vector Table (1/2)**

| Vector Table Address | Interrupt Source | 64-pin | 48-pin | 32-pin | 25-pin |
|---|---|---|---|---|---|
| 0000H | RESET, POR, LVD, WDT, TRAP, IAW, RPE | √ | √ | √ | √ |
| 0004H | INTWDTI/INTSRO | √ | √ | √ | √ |
| 0006H | INTLVI | √ | √ | √ | √ |
| 0008H | INTP0 | √ | √ | √ | √ |
| 000AH | INTP1 | √ | √ | √ | √ |
| 000CH | INTP2 | √ | √ | √ | √ |
| 000EH | INTP3 | √ | √ | √ | √ |
| 0010H | INTP4 | √ | √ | √ | √ |
| 0012H | INTP5 | √ | √ | – | – |
| 0014H | INTST2/INTCSI20/INTIIC20 | √ | √ | √ | – |
| 0016H | INTSR2/INTCSI21/INTIIC21 | √ | √ | Note 1 | – |
| 0018H | INTSRE2 | √ | √ | √ | – |
| 001AH | INTDMA0 | √ | √ | √ | √ |
| 001CH | INTDMA1 | √ | √ | √ | √ |
| 001EH | INTST0/INTCSI00/INTIIC00 | √ | √ | √ | √ |
| 0020H | INTSR0/INTCSI01/INTIIC01 | √ | √ | Note 2 | Note 2 |
| 0022H | INTSRE0 | √ | √ | √ | √ |
| | INTTM01H | √ | √ | √ | √ |
| 0024H | INTST1/INTCSI10/INTIIC10 | √ | Note 3 | Note 3 | Note 3 |
| 0026H | INTSR1/INTCSI11/INTIIC11 | √ | √ | √ | √ |
| 0028H | INTSRE1 | √ | √ | √ | √ |
| | INTTM03H | √ | √ | √ | √ |
| 002AH | INTIICA0 | √ | √ | √ | √ |
| 002CH | INTTM00 | √ | √ | √ | √ |
| 002EH | INTTM01 | √ | √ | √ | √ |
| 0030H | INTTM02 | √ | √ | √ | √ |
| 0032H | INTTM03 | √ | √ | √ | √ |
| 0034H | INTAD | √ | √ | √ | √ |
| 0036H | INTRTC | √ | √ | √ | √ |
| 0038H | INTIT | √ | √ | √ | √ |
| 003AH | INTKR | √ | √ | √ | Note 4 |

**Notes 1.** INTSR2 only.

**2.** INTSR0 only.

**3.** INTSR1 only.

**4.** When setting the peripheral I/O redirection register (PIOR).

**Table 3-3. Vector Table (2/2)**

| Vector Table Address | Interrupt Source | 64-pin | 48-pin | 32-pin | 25-pin |
|---|---|---|---|---|---|
| 0042H | INTTM04 | √ | √ | √ | √ |
| 0044H | INTTM05 | √ | √ | √ | √ |
| 0046H | INTTM06 | √ | √ | √ | √ |
| 0048H | INTTM07 | √ | √ | √ | √ |
| 004AH | INTP6 | √ | √ | – | – |
| 004CH | INTP7 | √ | – | – | – |
| 004EH | INTP8 | √ | √ | – | – |
| 0050H | INTP9 | √ | √ | – | – |
| 0052H | INTP10 | √ | – | – | – |
| 0054H | INTP11 | √ | – | – | – |
| 005EH | INTMD | √ | √ | √ | √ |
| 0062H | INTFL | √ | √ | √ | √ |
| 007EH | BRK | √ | √ | √ | √ |

**(2) CALLT instruction table area**

The 64-byte area 00080H to 000BFH can store the subroutine entry address of a 2-byte call instruction (CALLT). Set the subroutine entry address to a value in a range of 00000H to 0FFFFH (because an address code is of 2 bytes).
To use the boot swap function, set a CALLT instruction table also at 01080H to 010BFH.

**(3) Option byte area**

A 4-byte area of 000C0H to 000C3H can be used as an option byte area. Set the option byte at 010C0H to 010C3H when the boot swap is used. For details, see **CHAPTER 24 OPTION BYTE**.

**(4) On-chip debug security ID setting area**

A 10-byte area of 000C4H to 000CDH and 010C4H to 010CDH can be used as an on-chip debug security ID setting area. Set the on-chip debug security ID of 10 bytes at 000C4H to 000CDH when the boot swap is not used and at 000C4H to 000CDH and 010C4H to 010CDH when the boot swap is used. For details, see **CHAPTER 26 ON-CHIP DEBUG FUNCTION**.

### 3.1.2 Mirror area

<R> The RL78/G1A mirrors the code flash area of 00000H to 0FFFFH, to F0000H to FFFFFH. The products with 96 KB or more flash memory mirror the code flash area of 00000H to 0FFFFH or 10000H to 1FFFFH, to F0000H to FFFFFH (the code flash area to be mirrored is set by the processor mode control register (PMC)).

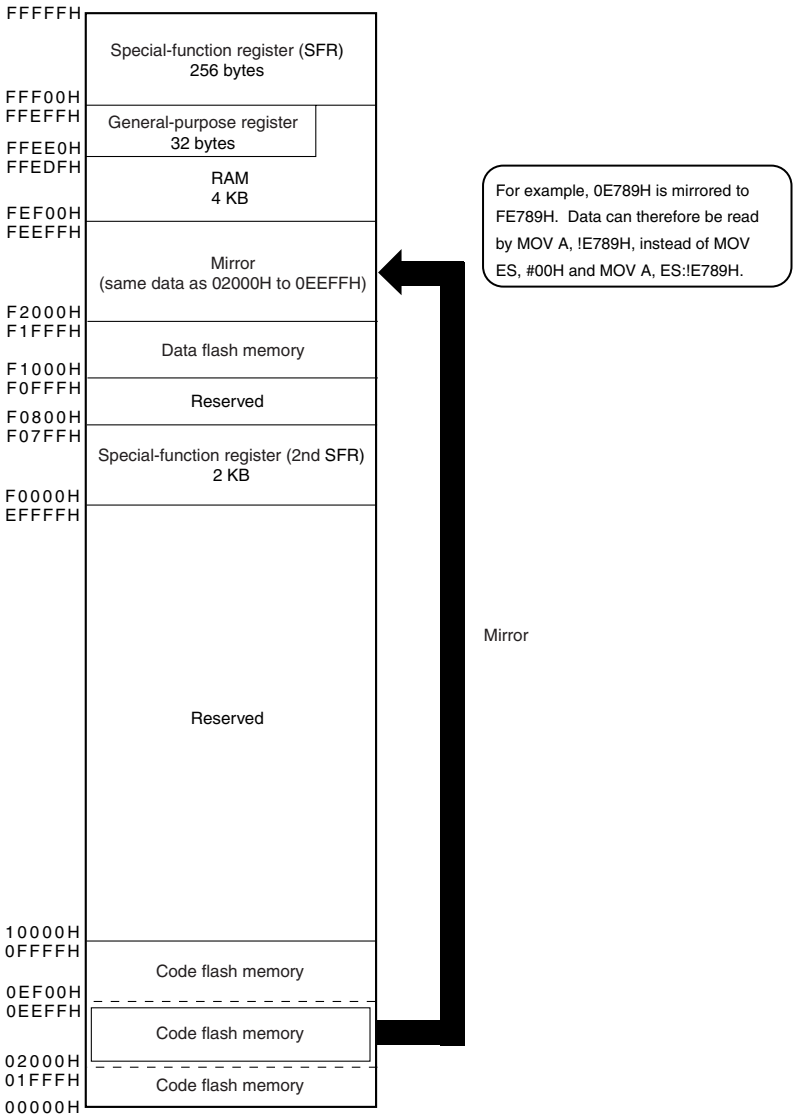
By reading data from F0000H to FFFFFH, an instruction that does not have the ES register as an operand can be used, and thus the contents of the code flash can be read with the shorter code. However, the code flash area is not mirrored to the SFR, extended SFR, RAM, and use prohibited areas.

See **3.1 Memory Space** for the mirror area of each product.

The mirror area can only be read and no instruction can be fetched from this area.

The following show examples.

**Example R5F10ExE (x = 8, B, G, L) (Flash memory: 64 KB, RAM: 4 KB)**




The PMC register is described below.

- **Processor mode control register (PMC)**

This register sets the flash memory space for mirroring to area from F0000H to FFFFFH.

The PMC register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

**Figure 3-5. Format of Processor Mode Control Register (PMC)**

Address: FFFFEH After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PMC | 0 | 0 | 0 | 0 | 0 | 0 | 0 | MAA |

| MAA | Selection of flash memory space for mirroring to area from F0000H to FFFFFH |
|---|---|
| 0 | 00000H to 0FFFFH is mirrored to F0000H to FFFFFH |
| 1 | Setting is prohibited |

**Cautions 1. Be sure to clear bit 0 (MAA) of this register to 0 (default value).**

**2. After setting the PMC register, wait for at least one instruction and access the mirror area.**

### 3.1.3 Internal data memory space

The RL78/G1A products incorporate the following RAMs.

**Table 3-4. Internal RAM Capacity**

| Part Number | Internal RAM |
|---|---|
| R5F10ExA (x = 8, B, G) | 2048 × 8 bits (FF700H to FFEFFH) |
| R5F10ExC (x = 8, B, G, L) | |
| R5F10ExD (x = 8, B, G, L) | 3072 × 8 bits (FF300H to FFEFFH) |
| R5F10ExE (x = 8, B, G, L) | 4096 × 8 bits (FEF00H to FFEFFH) |

The internal RAM can be used as a data area and a program area where instructions are fetched (it is prohibited to use the general-purpose register area for fetching instructions). Four general-purpose register banks consisting of eight 8-bit registers per bank are assigned to the 32-byte area of FFEE0H to FFEFFH of the internal RAM area.

The internal RAM is used as stack memory.

**Cautions 1. It is prohibited to use the general-purpose register (FFEE0H to FFEFFH) space for fetching instructions or as a stack area.**

**2. During self programming and data flash rewriting, the stack, data buffer, and RAM addresses used as branch destinations for vectored interrupts or as sources or destinations of DMA transfers must not be allocated to the area between addresses FFE20H and FFEDFH.**

**3. During self programming and data flash rewriting, the RAM area in the products below is prohibited. Because this area is used for each library.**

**R5F10ExD (x = 8, B, G, L): FF300H to FF309H**
**R5F10ExE (x = 8, B, G, L): FEF00H to FF309H**

### 3.1.4 Special function register (SFR) area

On-chip peripheral hardware special function registers (SFRs) are allocated in the area FFF00H to FFFFFH (see **Table 3-5** in **3.2.4 Special function registers (SFRs)**).

**Caution Do not access addresses to which SFRs are not assigned.**

### 3.1.5 Extended special function register (2nd SFR: 2nd Special Function Register) area

On-chip peripheral hardware special function registers (2nd SFRs) are allocated in the area F0000H to F07FFH (see **Table 3-6** in **3.2.5 Extended Special function registers (2nd SFRs: 2nd Special Function Registers)**).

SFRs other than those in the SFR area (FFF00H to FFFFFH) are allocated to this area. An instruction that accesses the extended SFR area, however, is 1 byte longer than an instruction that accesses the SFR area.

**Caution Do not access addresses to which extended SFRs are not assigned.**

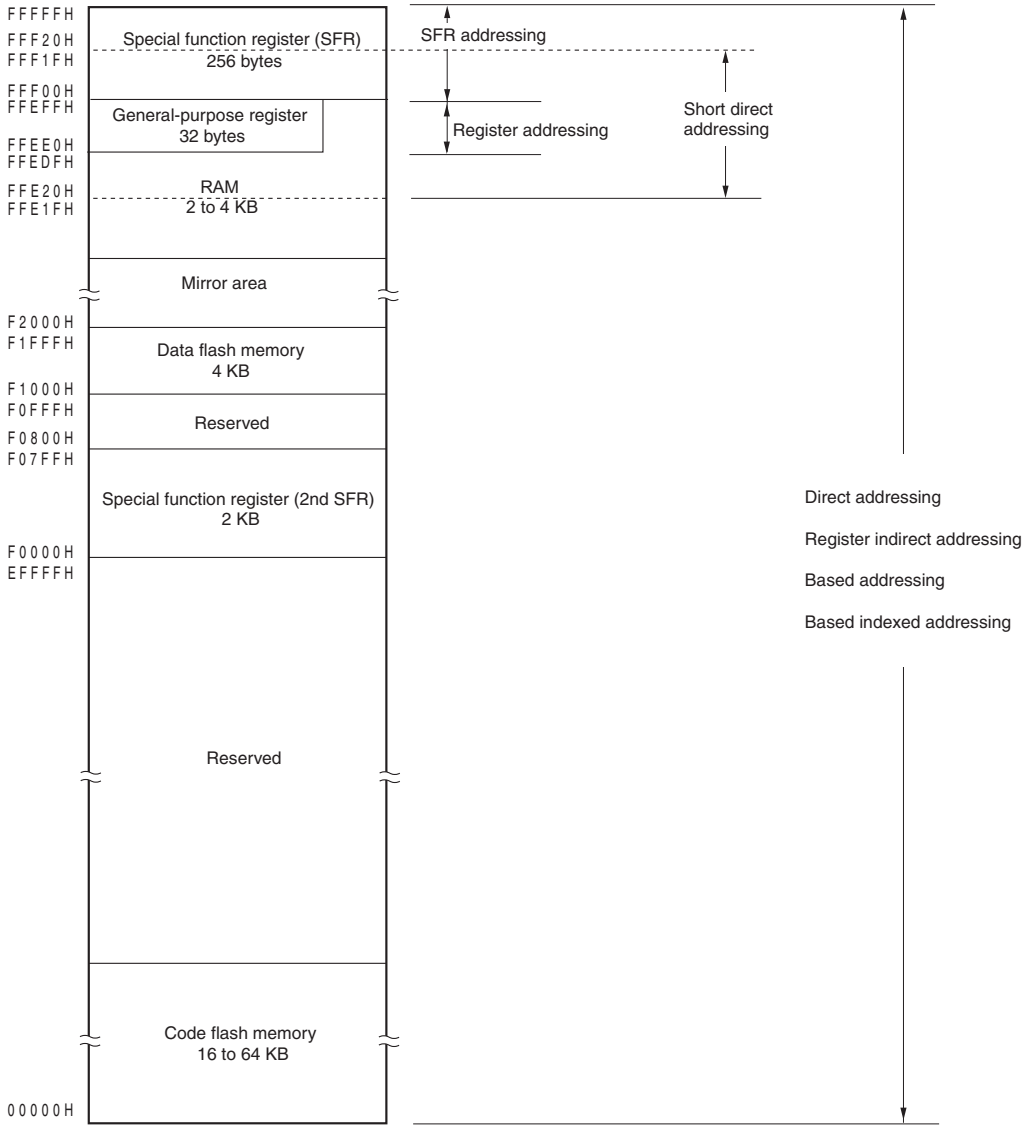
### 3.1.6 Data memory addressing

Addressing refers to the method of specifying the address of the instruction to be executed next or the address of the register or memory relevant to the execution of instructions.

Several addressing modes are provided for addressing the memory relevant to the execution of instructions for the RL78/G1A, based on operability and other considerations. In particular, special addressing methods designed for the functions of the special function registers (SFR) and general-purpose registers are available for use. Figure 3-6 shows correspondence between data memory and addressing. For details of each addressing, see **3.4 Addressing for Processing Data Addresses**.

<R>

**Figure 3-6. Correspondence Between Data Memory and Addressing**



## 3.2 Processor Registers

The RL78/G1A products incorporate the following processor registers.

### 3.2.1 Control registers

The control registers control the program sequence, statuses and stack memory. The control registers consist of a program counter (PC), a program status word (PSW) and a stack pointer (SP).

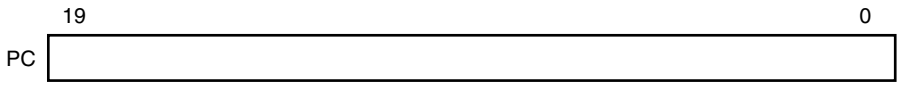
**(1) Program counter (PC)**

The program counter is a 20-bit register that holds the address information of the next program to be executed.

In normal operation, PC is automatically incremented according to the number of bytes of the instruction to be fetched. When a branch instruction is executed, immediate data and register contents are set.

Reset signal generation sets the reset vector table values at addresses 0000H and 0001H to the 16 lower-order bits of the program counter. The four higher-order bits of the program counter are cleared to 0000.

**Figure 3-7. Format of Program Counter**




**(2) Program status word (PSW)**

The program status word is an 8-bit register consisting of various flags set/reset by instruction execution.

Program status word contents are stored in the stack area upon vectored interrupt request is acknowledged or PUSH PSW instruction execution and are restored upon execution of the RETB, RETI and POP PSW instructions. Reset signal generation sets the PSW register to 06H.

**Figure 3-8. Format of Program Status Word**

| | 7 | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|
| PSW | IE | Z | RBS1 | AC | RBS0 | ISP1 | ISP0 | CY |

**(a) Interrupt enable flag (IE)**

This flag controls the interrupt request acknowledge operations of the CPU.

When 0, the IE flag is set to the interrupt disabled (DI) state, and all maskable interrupt requests are disabled.

When 1, the IE flag is set to the interrupt enabled (EI) state and maskable interrupt request acknowledgment is controlled with an in-service priority flag (ISP1, ISP0), an interrupt mask flag for various interrupt sources, and a priority specification flag.

The IE flag is reset (0) upon DI instruction execution or interrupt acknowledgment and is set (1) upon EI instruction execution.

**(b) Zero flag (Z)**

When the operation or comparison result is zero or equal, this flag is set (1). It is reset (0) in all other cases.

**(c) Register bank select flags (RBS0, RBS1)**

These are 2-bit flags to select one of the four register banks.

In these flags, the 2-bit information that indicates the register bank selected by SEL RBn instruction execution is stored.

**(d) Auxiliary carry flag (AC)**

If the operation result has a carry from bit 3 or a borrow at bit 3, this flag is set (1). It is reset (0) in all other cases.

**(e) In-service priority flags (ISP1, ISP0)**

This flag manages the priority of acknowledgeable maskable vectored interrupts. Vectored interrupt requests specified lower than the value of ISP0 and ISP1 flags by the priority specification flag registers (PRn0L, PRn0H, PRn1L, PRn1H, PRn2L, PRn2H) (see **16.3.3**) can not be acknowledged. Actual vectored interrupt request acknowledgment is controlled by the interrupt enable flag (IE).

**Remark** n = 0, 1

**(f) Carry flag (CY)**

This flag stores overflow and underflow upon add/subtract instruction execution. It stores the shift-out value upon rotate instruction execution and functions as a bit accumulator during bit operation instruction execution.

**(3) Stack pointer (SP)**

This is a 16-bit register to hold the start address of the memory stack area. Only the internal RAM area can be set as the stack area.

**Figure 3-9. Format of Stack Pointer**

<R>

| | 15 | | | | | | | | | | | | | | | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SP | SP15 | SP14 | SP13 | SP12 | SP11 | SP10 | SP9 | SP8 | SP7 | SP6 | SP5 | SP4 | SP3 | SP2 | SP1 | 0 |

In stack addressing through a stack pointer, the SP is decremented ahead of write (save) to the stack memory and is incremented after read (restore) from the stack memory.

**Cautions 1. Since reset signal generation makes the SP contents undefined, be sure to initialize the SP before using the stack.**

**2. It is prohibited to use the general-purpose register (FFEE0H to FFEFFH) space for fetching instruction or a stack area.**

<R> **3. Do not allocate RAM addresses which are used as a stack area, a data buffer, a branch destination of vector interrupt processing, and a DMA transfer destination/transfer source to the area FFE20H to FFEDFH when performing self-programming and rewriting the data flash memory.**

<R> **4. Use of the RAM areas of the following products is prohibited when performing self-programming and rewriting the data flash memory, because these areas are used for each library.**

**R5F10ExA (x = 8, B, G): FFE20H to FFEDFH**
**R5F10ExC (x = 8, B, G, L): FFE20H to FFEDFH**
**R5F10ExD (x = 8, B, G, L): FFE20H to FFEDFH, FF300H to FF309H**
**R5F10ExE (x = 8, B, G, L): FFE20H to FFEDFH, FEF00H to FF309H**

### 3.2.2 General-purpose registers

General-purpose registers are mapped at particular addresses (FFEE0H to FFEFFH) of the data memory. The general-purpose registers consists of 4 banks, each bank consisting of eight 8-bit registers (X, A, C, B, E, D, L, and H).

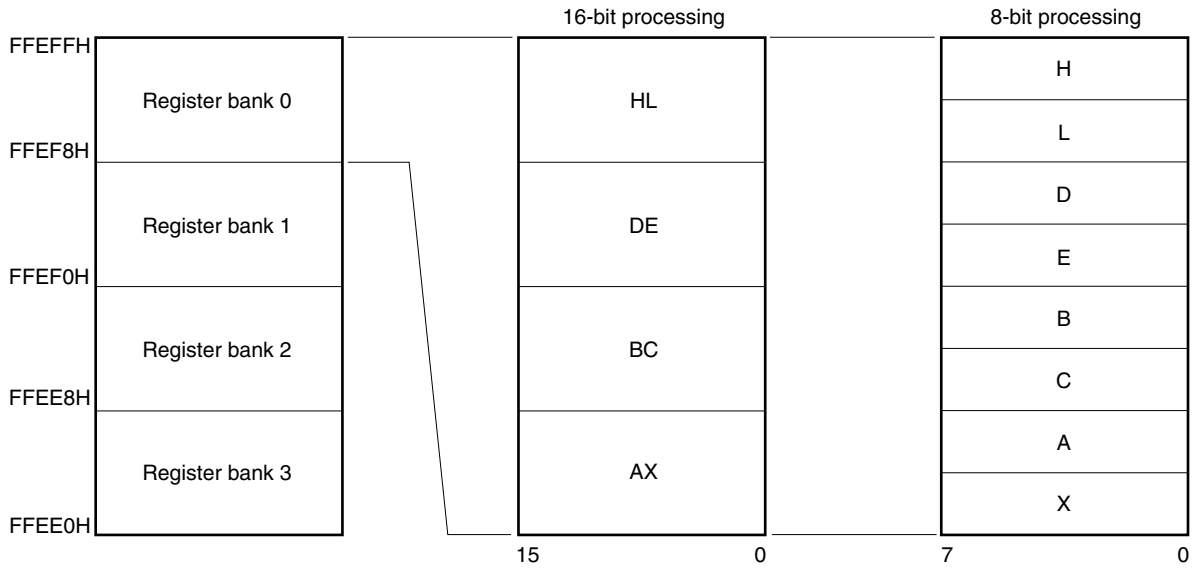
Each register can be used as an 8-bit register, and two 8-bit registers can also be used in a pair as a 16-bit register (AX, BC, DE, and HL).

Register banks to be used for instruction execution are set by the CPU control instruction (SEL RBn). Because of the 4-register bank configuration, an efficient program can be created by switching between a register for normal processing and a register for interrupt processing for each bank.

**Caution It is prohibited to use the general-purpose register (FFEE0H to FFEFFH) space for fetching instructions or as a stack area.**

**Figure 3-10. Configuration of General-Purpose Registers**

**(a) Function name**



### 3.2.3 ES and CS registers

The ES register and CS register are used to specify the higher address for data access and when a branch instruction is executed (register direct addressing), respectively.

The default value of the ES register after reset is 0FH, and that of the CS register is 00H.

**Figure 3-11. Configuration of ES and CS Registers**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ES | 0 | 0 | 0 | 0 | ES3 | ES2 | ES1 | ES0 |

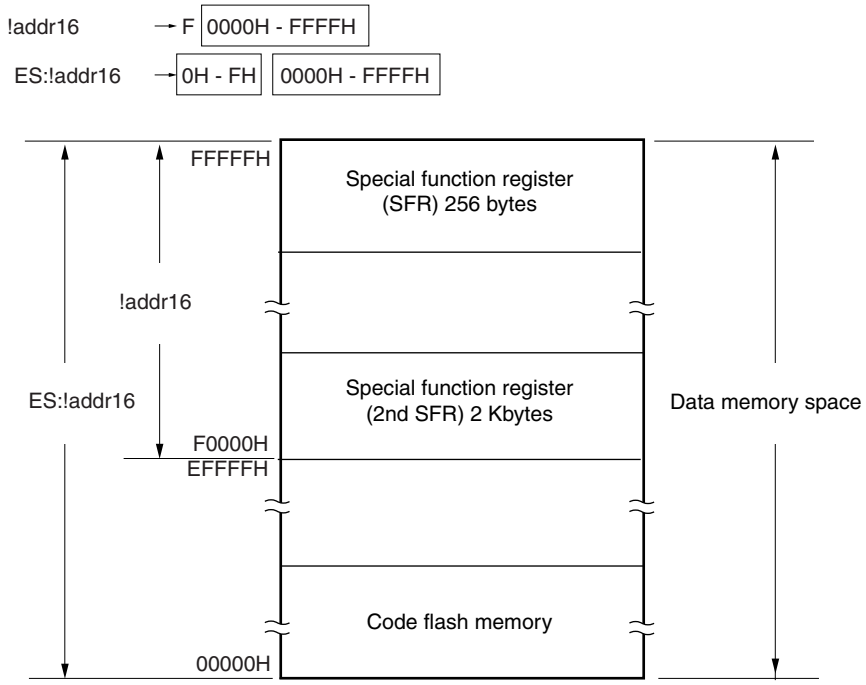
<R>

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CS | 0 | 0 | 0 | 0 | CS3 | CS2 | CS1 | CS0 |

Though the data area which can be accessed with 16-bit addresses is the 64 KB from F0000H to FFFFFH, using the ES register as well extends this to the 1 MB from 00000H to FFFFFH.

**Figure 3-12. Extension of Data Area Which Can Be Accessed**

<R>



### 3.2.4 Special function registers (SFRs)

Unlike a general-purpose register, each SFR has a special function.

SFRs are allocated to the FFF00H to FFFFFH area.

SFRs can be manipulated like general-purpose registers, using operation, transfer, and bit manipulation instructions. The manipulable bit units, 1, 8, and 16, depend on the SFR type.

Each manipulation bit unit can be specified as follows.

- 1-bit manipulation
  Describe as follows for the 1-bit manipulation instruction operand (sfr.bit).
  When the bit name is defined: <Bit name>
  When the bit name is not defined: <Register name>.<Bit number> or <Address>.<Bit number>
- 8-bit manipulation
  Describe the symbol defined by the assembler for the 8-bit manipulation instruction operand (sfr). This manipulation can also be specified with an address.
- 16-bit manipulation
  Describe the symbol defined by the assembler for the 16-bit manipulation instruction operand (sfrp). When specifying an address, describe an even address.

Table 3-5 gives a list of the SFRs. The meanings of items in the table are as follows.

- Symbol
  Symbol indicating the address of a special function register. It is a reserved word in the assembler, and is defined as an sfr variable using the #pragma sfr directive in the compiler. When using the assembler, debugger, and simulator, symbols can be written as an instruction operand.
- R/W
  Indicates whether the corresponding SFR can be read or written.
  R/W: Read/write enable
  R: Read only
  W: Write only
- Manipulable bit units
  "√" indicates the manipulable bit unit (1, 8, or 16). "–" indicates a bit unit for which manipulation is not possible.
- After reset
  Indicates each register status upon reset signal generation.

**Caution Do not access addresses to which extended SFRs are not assigned.**

**Remark** For extended SFRs (2nd SFRs), see **3.2.5 Extended special function registers (2nd SFRs: 2nd Special Function Registers)**.

**Table 3-5. SFR List (1/5)**

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|---|
| | | | | | 1-bit | 8-bit | 16-bit | |
| FFF00H | Port register 0 | P0 | | R/W | √ | √ | – | 00H |
| FFF01H | Port register 1 | P1 | | R/W | √ | √ | – | 00H |
| FFF02H | Port register 2 | P2 | | R/W | √ | √ | – | 00H |
| FFF03H | Port register 3 | P3 | | R/W | √ | √ | – | 00H |
| FFF04H | Port register 4 | P4 | | R/W | √ | √ | – | 00H |
| FFF05H | Port register 5 | P5 | | R/W | √ | √ | – | 00H |
| FFF06H | Port register 6 | P6 | | R/W | √ | √ | – | 00H |
| FFF07H | Port register 7 | P7 | | R/W | √ | √ | – | 00H |
| FFF0CH | Port register 12 | P12 | | R/W | √ | √ | – | Undefined |
| FFF0DH | Port register 13 | P13 | | R/W | √ | √ | – | Undefined |
| FFF0EH | Port register 14 | P14 | | R/W | √ | √ | – | 00H |
| FFF0FH | Port register 15 | P15 | | R/W | √ | √ | – | 00H |
| FFF10H | Serial data register 00 | TXD0/ SIO00 | SDR00 | R/W | – | √ | √ | 0000H |
| FFF11H | | – | | | – | – | | |
| FFF12H | Serial data register 01 | RXD0/ SIO01 | SDR01 | R/W | – | √ | √ | 0000H |
| FFF13H | | – | | | – | – | | |
| FFF18H | Timer data register 00 | TDR00 | | R/W | – | – | √ | 0000H |
| FFF19H | | | | | | | | |
| FFF1AH | Timer data register 01 | TDR01L | TDR01 | R/W | – | √ | √ | 00H |
| FFF1BH | | TDR01H | | | – | √ | | 00H |
| FFF1EH | 12-bit A/D conversion result register | ADCR | | R | – | – | √ | 0000H |
| FFF1FH | 8-bit A/D conversion result register | ADCRH | | R | – | √ | – | 00H |
| FFF20H | Port mode register 0 | PM0 | | R/W | √ | √ | – | FFH |
| FFF21H | Port mode register 1 | PM1 | | R/W | √ | √ | – | FFH |
| FFF22H | Port mode register 2 | PM2 | | R/W | √ | √ | – | FFH |
| FFF23H | Port mode register 3 | PM3 | | R/W | √ | √ | – | FFH |
| FFF24H | Port mode register 4 | PM4 | | R/W | √ | √ | – | FFH |
| FFF25H | Port mode register 5 | PM5 | | R/W | √ | √ | – | FFH |
| FFF26H | Port mode register 6 | PM6 | | R/W | √ | √ | – | FFH |
| FFF27H | Port mode register 7 | PM7 | | R/W | √ | √ | – | FFH |
| FFF2CH | Port mode register 12 | PM12 | | R/W | √ | √ | – | FFH |
| FFF2EH | Port mode register 14 | PM14 | | R/W | √ | √ | – | FFH |
| FFF2FH | Port mode register 15 | PM15 | | R/W | √ | √ | – | FFH |
| FFF30H | A/D converter mode register 0 | ADM0 | | R/W | √ | √ | – | 00H |
| FFF31H | Analog input channel specification register | ADS | | R/W | √ | √ | – | 00H |
| FFF32H | A/D converter mode register 1 | ADM1 | | R/W | √ | √ | – | 00H |

**Table 3-5. SFR List (2/5)**

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|---|
| | | | | | 1-bit | 8-bit | 16-bit | |
| FFF34H | Key return control register | KRCTL | | R/W | √ | √ | – | 00H |
| FFF35H | Key return flag register | KRF | | R/W | √ | √ | – | 00H |
| FFF36H | Key return mode control register 1 | KRM1 | | R/W | √ | √ | – | 00H |
| FFF37H | Key return mode control register 0 | KRM0 | | R/W | √ | √ | – | 00H |
| FFF38H | External interrupt rising edge enable register 0 | EGP0 | | R/W | √ | √ | – | 00H |
| FFF39H | External interrupt falling edge enable register 0 | EGN0 | | R/W | √ | √ | – | 00H |
| FFF3AH | External interrupt rising edge enable register 1 | EGP1 | | R/W | √ | √ | – | 00H |
| FFF3BH | External interrupt falling edge enable register 1 | EGN1 | | R/W | √ | √ | – | 00H |
| FFF44H | Serial data register 02 | TXD1/ SIO10 | SDR02 | R/W | – | √ | √ | 0000H |
| FFF45H | | – | | | – | – | | |
| FFF46H | Serial data register 03 | RXD1/ SIO11 | SDR03 | R/W | – | √ | √ | 0000H |
| FFF47H | | – | | | – | – | | |
| FFF48H | Serial data register 10 | TXD2/ SIO20 | SDR10 | R/W | – | √ | √ | 0000H |
| FFF49H | | – | | | – | – | | |
| FFF4AH | Serial data register 11 | RXD2/ SIO21 | SDR11 | R/W | – | √ | √ | 0000H |
| FFF4BH | | – | | | – | – | | |
| FFF50H | IICA shift register 0 | IICA0 | | R/W | – | √ | – | 00H |
| FFF51H | IICA status register 0 | IICS0 | | R | √ | √ | – | 00H |
| FFF52H | IICA flag register 0 | IICF0 | | R/W | √ | √ | – | 00H |
| FFF64H | Timer data register 02 | TDR02 | | R/W | – | – | √ | 0000H |
| FFF65H | | | | | | | | |
| FFF66H | Timer data register 03 | TDR03L | TDR03 | R/W | – | √ | √ | 00H |
| FFF67H | | TDR03H | | | – | √ | | 00H |
| FFF68H | Timer data register 04 | TDR04 | | R/W | – | – | √ | 0000H |
| FFF69H | | | | | | | | |
| FFF6AH | Timer data register 05 | TDR05 | | R/W | – | – | √ | 0000H |
| FFF6BH | | | | | | | | |
| FFF6CH | Timer data register 06 | TDR06 | | R/W | – | – | √ | 0000H |
| FFF6DH | | | | | | | | |
| FFF6EH | Timer data register 07 | TDR07 | | R/W | – | – | √ | 0000H |
| FFF6FH | | | | | | | | |

**Table 3-5. SFR List (3/5)**

| Address | Special Function Register (SFR) Name | Symbol | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|
| | | | | 1-bit | 8-bit | 16-bit | |
| FFF90H<br>FFF91H | Interval timer control register | ITMC | R/W | – | – | √ | 0FFFH |
| FFF92H | Second count register | SEC | R/W | – | √ | – | 00H |
| FFF93H | Minute count register | MIN | R/W | – | √ | – | 00H |
| FFF94H | Hour count register | HOUR | R/W | – | √ | – | 12H[Note] |
| FFF95H | Week count register | WEEK | R/W | – | √ | – | 00H |
| FFF96H | Day count register | DAY | R/W | – | √ | – | 01H |
| FFF97H | Month count register | MONTH | R/W | – | √ | – | 01H |
| FFF98H | Year count register | YEAR | R/W | – | √ | – | 00H |
| FFF99H | Watch error correction register | SUBCUD | R/W | – | √ | – | 00H |
| FFF9AH | Alarm minute register | ALARMWM | R/W | – | √ | – | 00H |
| FFF9BH | Alarm hour register | ALARMWH | R/W | – | √ | – | 12H |
| FFF9CH | Alarm week register | ALARMWW | R/W | – | √ | – | 00H |
| FFF9DH | Real-time clock control register 0 | RTCC0 | R/W | √ | √ | – | 00H |
| FFF9EH | Real-time clock control register 1 | RTCC1 | R/W | √ | √ | – | 00H |
| FFFA0H | Clock operation mode control register | CMC | R/W | – | √ | – | 00H |
| FFFA1H | Clock operation status control register | CSC | R/W | √ | √ | – | C0H |
| FFFA2H | Oscillation stabilization time counter status register | OSTC | R | √ | √ | – | 00H |
| FFFA3H | Oscillation stabilization time select register | OSTS | R/W | – | √ | – | 07H |
| FFFA4H | System clock control register | CKC | R/W | √ | √ | – | 00H |
| FFFA5H | Clock output select register 0 | CKS0 | R/W | √ | √ | – | 00H |
| FFFA6H | Clock output select register 1 | CKS1 | R/W | √ | √ | – | 00H |

**Note** The value of this register is 00H if the AMPM bit (bit 3 of real-time clock control register 0 (RTCC0)) is set to 1 after reset.

**Table 3-5. SFR List (4/5)**

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|---|
| | | | | | 1-bit | 8-bit | 16-bit | |
| FFFA8H | Reset control flag register | RESF | | R | – | √ | – | Undefined Note 1 |
| FFFA9H | Voltage detection register | LVIM | | R/W | √ | √ | – | 00H Note 1 |
| FFFAAH | Voltage detection level register | LVIS | | R/W | √ | √ | – | 00H/01H/81H Note 1 |
| FFFABH | Watchdog timer enable register | WDTE | | R/W | – | √ | – | 1AH/9AH Note 2 |
| FFFACH | CRC input register | CRCIN | | R/W | – | √ | – | 00H |
| FFFB0H | DMA SFR address register 0 | DSA0 | | R/W | – | √ | – | 00H |
| FFFB1H | DMA SFR address register 1 | DSA1 | | R/W | – | √ | – | 00H |
| FFFB2H | DMA RAM address register 0 | DRA0L | DRA0 | R/W | – | √ | √ | 00H |
| FFFB3H | | DRA0H | | R/W | – | √ | | 00H |
| FFFB4H | DMA RAM address register 1 | DRA1L | DRA1 | R/W | – | √ | √ | 00H |
| FFFB5H | | DRA1H | | R/W | – | √ | | 00H |
| FFFB6H | DMA byte count register 0 | DBC0L | DBC0 | R/W | – | √ | √ | 00H |
| FFFB7H | | DBC0H | | R/W | – | √ | | 00H |
| FFFB8H | DMA byte count register 1 | DBC1L | DBC1 | R/W | – | √ | √ | 00H |
| FFFB9H | | DBC1H | | R/W | – | √ | | 00H |
| FFFBAH | DMA mode control register 0 | DMC0 | | R/W | √ | √ | – | 00H |
| FFFBBH | DMA mode control register 1 | DMC1 | | R/W | √ | √ | – | 00H |
| FFFBCH | DMA operation control register 0 | DRC0 | | R/W | √ | √ | – | 00H |
| FFFBDH | DMA operation control register 1 | DRC1 | | R/W | √ | √ | – | 00H |

<R> **Notes 1.** The reset values of the registers vary depending on the reset source as shown below.

<R>

| Register \ Reset Source | | $\overline{\text{RESET}}$ Input | Reset by POR | Reset by Execution of Illegal Instruction | Reset by WDT | Reset by RAM parity error | Reset by illegal-memory access | Reset by LVD |
|---|---|---|---|---|---|---|---|---|
| RESF | TRAP | Cleared (0) | | Set (1) | Held | | | Held |
| | WDTRF | | | Held | Set (1) | Held | | |
| | RPERF | | | Held | | Set (1) | Held | |
| | IAWRF | | | Held | | | Set (1) | |
| | LVIRF | | | Held | | | | Set (1) |
| LVIM | LVISEN | Cleared (0) | | | | | | Held |
| | LVIOMSK | Held | | | | | | |
| | LVIF | | | | | | | |
| LVIS | | Cleared (00H/01H/81H) | | | | | | |

**2.** The reset value of the WDTE register is determined by the setting of the option byte.

**Table 3-5. SFR List (5/5)**

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|---|
| | | | | | 1-bit | 8-bit | 16-bit | |
| FFFD0H | Interrupt request flag register 2 | IF2L | IF2 | R/W | √ | √ | √ | 00H |
| FFFD1H | | IF2H | | R/W | √ | √ | | 00H |
| FFFD4H | Interrupt mask flag register 2 | MK2L | MK2 | R/W | √ | √ | √ | FFH |
| FFFD5H | | MK2H | | R/W | √ | √ | | FFH |
| FFFD8H | Priority specification flag register 02 | PR02L | PR02 | R/W | √ | √ | √ | FFH |
| FFFD9H | | PR02H | | R/W | √ | √ | | FFH |
| FFFDCH | Priority specification flag register 12 | PR12L | PR12 | R/W | √ | √ | √ | FFH |
| FFFDDH | | PR12H | | R/W | √ | √ | | FFH |
| FFFE0H | Interrupt request flag register 0 | IF0L | IF0 | R/W | √ | √ | √ | 00H |
| FFFE1H | | IF0H | | R/W | √ | √ | | 00H |
| FFFE2H | Interrupt request flag register 1 | IF1L | IF1 | R/W | √ | √ | √ | 00H |
| FFFE3H | | IF1H | | R/W | √ | √ | | 00H |
| FFFE4H | Interrupt mask flag register 0 | MK0L | MK0 | R/W | √ | √ | √ | FFH |
| FFFE5H | | MK0H | | R/W | √ | √ | | FFH |
| FFFE6H | Interrupt mask flag register 1 | MK1L | MK1 | R/W | √ | √ | √ | FFH |
| FFFE7H | | MK1H | | R/W | √ | √ | | FFH |
| FFFE8H | Priority specification flag register 00 | PR00L | PR00 | R/W | √ | √ | √ | FFH |
| FFFE9H | | PR00H | | R/W | √ | √ | | FFH |
| FFFEAH | Priority specification flag register 01 | PR01L | PR01 | R/W | √ | √ | √ | FFH |
| FFFEBH | | PR01H | | R/W | √ | √ | | FFH |
| FFFECH | Priority specification flag register 10 | PR10L | PR10 | R/W | √ | √ | √ | FFH |
| FFFEDH | | PR10H | | R/W | √ | √ | | FFH |
| FFFEEH | Priority specification flag register 11 | PR11L | PR11 | R/W | √ | √ | √ | FFH |
| FFFEFH | | PR11H | | R/W | √ | √ | | FFH |
| FFFF0H | Multiplication/division data register A (L) | MDAL | | R/W | – | – | √ | 0000H |
| FFFF1H | | | | | | | | |
| FFFF2H | Multiplication/division data register A (H) | MDAH | | R/W | – | – | √ | 0000H |
| FFFF3H | | | | | | | | |
| FFFF4H | Multiplication/division data register B (H) | MDBH | | R/W | – | – | √ | 0000H |
| FFFF5H | | | | | | | | |
| FFFF6H | Multiplication/division data register B (L) | MDBL | | R/W | – | – | √ | 0000H |
| FFFF7H | | | | | | | | |
| FFFFEH | Processor mode control register | PMC | | R/W | √ | √ | – | 00H |

**Remark** For extended SFRs (2nd SFRs), see **Table 3-6 Extended SFR (2nd SFR) List**.

### 3.2.5 Extended special function registers (2nd SFRs)

Unlike a general-purpose register, each extended SFR (2nd SFR) has a special function.

Extended SFRs are allocated to the F0000H to F07FFH area. SFRs other than those in the SFR area (FFF00H to FFFFFH) are allocated to this area. An instruction that accesses the extended SFR area, however, is 1 byte longer than an instruction that accesses the SFR area.

Extended SFRs can be manipulated like general-purpose registers, using operation, transfer, and bit manipulation instructions. The manipulable bit units, 1, 8, and 16, depend on the SFR type.

Each manipulation bit unit can be specified as follows.

- 1-bit manipulation
  Describe as follows for the 1-bit manipulation instruction operand (!addr16.bit)
  When the bit name is defined: <Bit name>
  When the bit name is not defined: <Register name>.<Bit number> or <Address>.<Bit number>
- 8-bit manipulation
  Describe the symbol defined by the assembler for the 8-bit manipulation instruction operand (!addr16). This manipulation can also be specified with an address.
- 16-bit manipulation
  Describe the symbol defined by the assembler for the 16-bit manipulation instruction operand (!addr16). When specifying an address, describe an even address.

Table 3-6 gives a list of the extended SFRs. The meanings of items in the table are as follows.

- Symbol
  Symbol indicating the address of an extended SFR. It is a reserved word in the assembler, and is defined as an sfr variable using the #pragma sfr directive in the compiler. When using the assembler, debugger, and simulator, symbols can be written as an instruction operand.
- R/W
  Indicates whether the corresponding extended SFR can be read or written.
  R/W: Read/write enable
  R: Read only
  W: Write only
- Manipulable bit units
  "√" indicates the manipulable bit unit (1, 8, or 16). "–" indicates a bit unit for which manipulation is not possible.
- After reset
  Indicates each register status upon reset signal generation.

**Caution Do not access addresses to which extended SFRs (2nd SFRs) are not assigned.**

**Remark** For SFRs in the SFR area, see **3.2.4 Special function registers (SFRs)**.

**Table 3-6. Extended SFR (2nd SFR) List (1/6)**

| Address | Special Function Register (SFR) Name | Symbol | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|
| | | | | 1-bit | 8-bit | 16-bit | |
| F0010H | A/D converter mode register 2 | ADM2 | R/W | √ | √ | – | 00H |
| F0011H | Conversion result comparison upper limit setting register | ADUL | R/W | – | √ | – | FFH |
| F0012H | Conversion result comparison lower limit setting register | ADLL | R/W | – | √ | – | 00H |
| F0013H | A/D test register | ADTES | R/W | – | √ | – | 00H |
| F0030H | Pull-up resistor option register 0 | PU0 | R/W | √ | √ | – | 00H |
| F0031H | Pull-up resistor option register 1 | PU1 | R/W | √ | √ | – | 00H |
| F0033H | Pull-up resistor option register 3 | PU3 | R/W | √ | √ | – | 00H |
| F0034H | Pull-up resistor option register 4 | PU4 | R/W | √ | √ | – | 01H |
| F0035H | Pull-up resistor option register 5 | PU5 | R/W | √ | √ | – | 00H |
| F0037H | Pull-up resistor option register 7 | PU7 | R/W | √ | √ | – | 00H |
| F003CH | Pull-up resistor option register 12 | PU12 | R/W | √ | √ | – | 00H |
| F003EH | Pull-up resistor option register 14 | PU14 | R/W | √ | √ | – | 00H |
| F0040H | Port input mode register 0 | PIM0 | R/W | √ | √ | – | 00H |
| F0041H | Port input mode register 1 | PIM1 | R/W | √ | √ | – | 00H |
| F0050H | Port output mode register 0 | POM0 | R/W | √ | √ | – | 00H |
| F0051H | Port output mode register 1 | POM1 | R/W | √ | √ | – | 00H |
| F0055H | Port output mode register 5 | POM5 | R/W | √ | √ | – | 00H |
| F0057H | Port output mode register 7 | POM7 | R/W | √ | √ | – | 00H |
| F0060H | Port mode control register 0 | PMC0 | R/W | √ | √ | – | FFH |
| F0061H | Port mode control register 1 | PMC1 | R/W | √ | √ | – | FFH |
| F0063H | Port mode control register 3 | PMC3 | R/W | √ | √ | – | FFH |
| F0064H | Port mode control register 4 | PMC4 | R/W | √ | √ | – | FFH |
| F0065H | Port mode control register 5 | PMC5 | R/W | √ | √ | – | FFH |
| F0067H | Port mode control register 7 | PMC7 | R/W | √ | √ | – | FFH |
| F006CH | Port mode control register 12 | PMC12 | R/W | √ | √ | – | FFH |
| F0070H | Noise filter enable register 0 | NFEN0 | R/W | √ | √ | – | 00H |
| F0071H | Noise filter enable register 1 | NFEN1 | R/W | √ | √ | – | 00H |
| F0073H | Input switch control register | ISC | R/W | √ | √ | – | 00H |
| F0074H | Timer input select register 0 | TIS0 | R/W | – | √ | – | 00H |
| F0076H | A/D port configuration register | ADPC | R/W | – | √ | – | 00H |
| F0077H | Peripheral I/O redirection register | PIOR | R/W | – | √ | – | 00H |
| F0078H | Invalid memory access detection control register | IAWCTL | R/W | – | √ | – | 00H |

**Table 3-6. Extended SFR (2nd SFR) List (2/6)**

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|---|
| | | | | | 1-bit | 8-bit | 16-bit | |
| F007CH | Global analog input disable register | GAIDIS | | R/W | √ | √ | – | 00H |
| F007DH | Global digital input disable register | GDIDIS | | R/W | √ | √ | – | 00H |
| F0090H | Data flash control register | DFLCTL | | R/W | √ | √ | – | 00H |
| F00A0H | High-speed on-chip oscillator trimming register | HIOTRM | | R/W | – | √ | – | Undefined Note 1 |
| <R> F00A8H | High-speed on-chip oscillator frequency select register | HOCODIV | | R/W | – | √ | – | Undefined Note 2 |
| F00E0H | Multiplication/division data register C (L) | MDCL | | R/W | – | – | √ | 0000H |
| F00E2H | Multiplication/division data register C (H) | MDCH | | R/W | – | – | √ | 0000H |
| F00E8H | Multiplication/division control register | MDUC | | R/W | √ | √ | – | 00H |
| F00F0H | Peripheral enable register 0 | PER0 | | R/W | √ | √ | – | 00H |
| F00F3H | Subsystem clock supply mode control register | OSMC | | R/W | – | √ | – | 00H |
| F00F5H | RAM parity error control register | RPECTL | | R/W | √ | √ | – | 00H |
| F00FEH | BCD adjust result register | BCDADJ | | R | – | √ | – | Undefined |
| F0100H | Serial status register 00 | SSR00L | SSR00 | R | – | √ | √ | 0000H |
| F0101H | | – | | | – | – | | |
| F0102H | Serial status register 01 | SSR01L | SSR01 | R | – | √ | √ | 0000H |
| F0103H | | – | | | – | – | | |
| F0104H | Serial status register 02 | SSR02L | SSR02 | R | – | √ | √ | 0000H |
| F0105H | | – | | | – | – | | |
| F0106H | Serial status register 03 | SSR03L | SSR03 | R | – | √ | √ | 0000H |
| F0107H | | – | | | – | – | | |
| F0108H | Serial flag clear trigger register 00 | SIR00L | SIR00 | R/W | – | √ | √ | 0000H |
| F0109H | | – | | | – | – | | |
| F010AH | Serial flag clear trigger register 01 | SIR01L | SIR01 | R/W | – | √ | √ | 0000H |
| F010BH | | – | | | – | – | | |
| F010CH | Serial flag clear trigger register 02 | SIR02L | SIR02 | R/W | – | √ | √ | 0000H |
| F010DH | | – | | | – | – | | |
| F010EH | Serial flag clear trigger register 03 | SIR03L | SIR03 | R/W | – | √ | √ | 0000H |
| F010FH | | – | | | – | – | | |

**Notes 1.** The value after a reset is adjusted at the time of shipment.

<R> **2.** The value after a reset is a value set in FRQSEL2 to FRQSEL0 of the option byte (000C2H).

**Table 3-6. Extended SFR (2nd SFR) List (3/6)**

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|---|
| | | | | | 1-bit | 8-bit | 16-bit | |
| F0110H | Serial mode register 00 | SMR00 | | R/W | – | – | √ | 0020H |
| F0111H | | | | | | | | |
| F0112H | Serial mode register 01 | SMR01 | | R/W | – | – | √ | 0020H |
| F0113H | | | | | | | | |
| F0114H | Serial mode register 02 | SMR02 | | R/W | – | – | √ | 0020H |
| F0115H | | | | | | | | |
| F0116H | Serial mode register 03 | SMR03 | | R/W | – | – | √ | 0020H |
| F0117H | | | | | | | | |
| F0118H | Serial communication operation setting register 00 | SCR00 | | R/W | – | – | √ | 0087H |
| F0119H | | | | | | | | |
| F011AH | Serial communication operation setting register 01 | SCR01 | | R/W | – | – | √ | 0087H |
| F011BH | | | | | | | | |
| F011CH | Serial communication operation setting register 02 | SCR02 | | R/W | – | – | √ | 0087H |
| F011DH | | | | | | | | |
| F011EH | Serial communication operation setting register 03 | SCR03 | | R/W | – | – | √ | 0087H |
| F011FH | | | | | | | | |
| F0120H | Serial channel enable status register 0 | SE0L | SE0 | R | √ | √ | √ | 0000H |
| F0121H | | – | | | – | – | | |
| F0122H | Serial channel start register 0 | SS0L | SS0 | R/W | √ | √ | √ | 0000H |
| F0123H | | – | | | – | – | | |
| F0124H | Serial channel stop register 0 | ST0L | ST0 | R/W | √ | √ | √ | 0000H |
| F0125H | | – | | | – | – | | |
| F0126H | Serial clock select register 0 | SPS0L | SPS0 | R/W | – | √ | √ | 0000H |
| F0127H | | – | | | – | – | | |
| F0128H | Serial output register 0 | SO0 | | R/W | – | – | √ | 0F0FH |
| F0129H | | | | | | | | |
| F012AH | Serial output enable register 0 | SOE0L | SOE0 | R/W | √ | √ | √ | 0000H |
| F012BH | | – | | | – | – | | |
| F0134H | Serial output level register 0 | SOL0L | SOL0 | R/W | – | √ | √ | 0000H |
| F0135H | | – | | | – | – | | |
| F0138H | Serial standby control register 0 | SSC0L | SSC0 | R/W | – | √ | √ | 0000H |
| | | – | | | – | – | | |
| F0140H | Serial status register 10 | SSR10L | SSR10 | R | – | √ | √ | 0000H |
| F0141H | | – | | | – | – | | |
| F0142H | Serial status register 11 | SSR11L | SSR11 | R | – | √ | √ | 0000H |
| F0143H | | – | | | – | – | | |
| F0148H | Serial flag clear trigger register 10 | SIR10L | SIR10 | R/W | – | √ | √ | 0000H |
| F0149H | | – | | | – | – | | |
| F014AH | Serial flag clear trigger register 11 | SIR11L | SIR11 | R/W | – | √ | √ | 0000H |
| F014BH | | – | | | – | – | | |

**Table 3-6. Extended SFR (2nd SFR) List (4/6)**

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|---|
| | | | | | 1-bit | 8-bit | 16-bit | |
| F0150H | Serial mode register 10 | SMR10 | | R/W | – | – | √ | 0020H |
| F0151H | | | | | | | | |
| F0152H | Serial mode register 11 | SMR11 | | R/W | – | – | √ | 0020H |
| F0153H | | | | | | | | |
| F0158H | Serial communication operation setting register 10 | SCR10 | | R/W | – | – | √ | 0087H |
| F0159H | | | | | | | | |
| F015AH | Serial communication operation setting register 11 | SCR11 | | R/W | – | – | √ | 0087H |
| F015BH | | | | | | | | |
| F0160H | Serial channel enable status register 1 | SE1L | SE1 | R | √ | √ | √ | 0000H |
| F0161H | | – | | | – | – | | |
| F0162H | Serial channel start register 1 | SS1L | SS1 | R/W | √ | √ | √ | 0000H |
| F0163H | | – | | | – | – | | |
| F0164H | Serial channel stop register 1 | ST1L | ST1 | R/W | √ | √ | √ | 0000H |
| F0165H | | – | | | – | – | | |
| F0166H | Serial clock select register 1 | SPS1L | SPS1 | R/W | – | √ | √ | 0000H |
| F0167H | | – | | | – | – | | |
| F0168H | Serial output register 1 | SO1 | | R/W | – | – | √ | 0303H |
| F0169H | | | | | | | | |
| F016AH | Serial output enable register 1 | SOE1L | SOE1 | R/W | √ | √ | √ | 0000H |
| F016BH | | – | | | – | – | | |
| F0174H | Serial output level register 1 | SOL1L | SOL1 | R/W | – | √ | √ | 0000H |
| F0175H | | – | | | – | – | | |
| F0180H | Timer counter register 00 | TCR00 | | R | – | – | √ | FFFFH |
| F0181H | | | | | | | | |
| F0182H | Timer counter register 01 | TCR01 | | R | – | – | √ | FFFFH |
| F0183H | | | | | | | | |
| F0184H | Timer counter register 02 | TCR02 | | R | – | – | √ | FFFFH |
| F0185H | | | | | | | | |
| F0186H | Timer counter register 03 | TCR03 | | R | – | – | √ | FFFFH |
| F0187H | | | | | | | | |
| F0188H | Timer counter register 04 | TCR04 | | R | – | – | √ | FFFFH |
| F0189H | | | | | | | | |
| F018AH | Timer counter register 05 | TCR05 | | R | – | – | √ | FFFFH |
| F018BH | | | | | | | | |
| F018CH | Timer counter register 06 | TCR06 | | R | – | – | √ | FFFFH |
| F018DH | | | | | | | | |
| F018EH | Timer counter register 07 | TCR07 | | R | – | – | √ | FFFFH |
| F018FH | | | | | | | | |

**Table 3-6. Extended SFR (2nd SFR) List (5/6)**

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|---|
| | | | | | 1-bit | 8-bit | 16-bit | |
| F0190H | Timer mode register 00 | TMR00 | | R/W | – | – | √ | 0000H |
| F0191H | | | | | | | | |
| F0192H | Timer mode register 01 | TMR01 | | R/W | – | – | √ | 0000H |
| F0193H | | | | | | | | |
| F0194H | Timer mode register 02 | TMR02 | | R/W | – | – | √ | 0000H |
| F0195H | | | | | | | | |
| F0196H | Timer mode register 03 | TMR03 | | R/W | – | – | √ | 0000H |
| F0197H | | | | | | | | |
| F0198H | Timer mode register 04 | TMR04 | | R/W | – | – | √ | 0000H |
| F0199H | | | | | | | | |
| F019AH | Timer mode register 05 | TMR05 | | R/W | – | – | √ | 0000H |
| F019BH | | | | | | | | |
| F019CH | Timer mode register 06 | TMR06 | | R/W | – | – | √ | 0000H |
| F019DH | | | | | | | | |
| F019EH | Timer mode register 07 | TMR07 | | R/W | – | – | √ | 0000H |
| F019FH | | | | | | | | |
| F01A0H | Timer status register 00 | TSR00L | TSR00 | R | – | √ | √ | 0000H |
| F01A1H | | – | | | – | – | | |
| F01A2H | Timer status register 01 | TSR01L | TSR01 | R | – | √ | √ | 0000H |
| F01A3H | | – | | | – | – | | |
| F01A4H | Timer status register 02 | TSR02L | TSR02 | R | – | √ | √ | 0000H |
| F01A5H | | – | | | – | – | | |
| F01A6H | Timer status register 03 | TSR03L | TSR03 | R | – | √ | √ | 0000H |
| F01A7H | | – | | | – | – | | |
| F01A8H | Timer status register 04 | TSR04L | TSR04 | R | – | √ | √ | 0000H |
| F01A9H | | – | | | – | – | | |
| F01AAH | Timer status register 05 | TSR05L | TSR05 | R | – | √ | √ | 0000H |
| F01ABH | | – | | | – | – | | |
| F01ACH | Timer status register 06 | TSR06L | TSR06 | R | – | √ | √ | 0000H |
| F01ADH | | – | | | – | – | | |
| F01AEH | Timer status register 07 | TSR07L | TSR07 | R | – | √ | √ | 0000H |
| F01AFH | | – | | | – | – | | |

**Table 3-6. Extended SFR (2nd SFR) List (6/6)**

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|---|
| | | | | | 1-bit | 8-bit | 16-bit | |
| F01B0H | Timer channel enable status register 0 | TE0L | TE0 | R | √ | √ | √ | 0000H |
| F01B1H | | – | | | – | – | | |
| F01B2H | Timer channel start register 0 | TS0L | TS0 | R/W | √ | √ | √ | 0000H |
| F01B3H | | – | | | – | – | | |
| F01B4H | Timer channel stop register 0 | TT0L | TT0 | R/W | √ | √ | √ | 0000H |
| F01B5H | | – | | | – | – | | |
| F01B6H | Timer clock select register 0 | TPS0 | | R/W | – | – | √ | 0000H |
| F01B7H | | | | | | | | |
| F01B8H | Timer output register 0 | TO0L | TO0 | R/W | – | √ | √ | 0000H |
| F01B9H | | – | | | – | – | | |
| F01BAH | Timer output enable register 0 | TOE0L | TOE0 | R/W | √ | √ | √ | 0000H |
| F01BBH | | – | | | – | – | | |
| F01BCH | Timer output level register 0 | TOL0L | TOL0 | R/W | – | √ | √ | 0000H |
| F01BDH | | – | | | – | – | | |
| F01BEH | Timer output mode register 0 | TOM0L | TOM0 | R/W | – | √ | √ | 0000H |
| F01BFH | | – | | | – | – | | |
| F0230H | IICA control register 00 | IICCTL00 | R/W | √ | √ | – | 00H | F0230H |
| F0231H | IICA control register 01 | IICCTL01 | R/W | √ | √ | – | 00H | F0231H |
| F0232H | IICA low-level width setting register 0 | IICWL0 | R/W | – | √ | – | FFH | F0232H |
| F0233H | IICA high-level width setting register 0 | IICWH0 | R/W | – | √ | – | FFH | F0233H |
| F0234H | Slave address register 0 | SVA0 | R/W | – | √ | – | 00H | F0234H |
| F02F0H | Flash memory CRC control register | CRC0CTL | R/W | √ | √ | – | 00H | F02F0H |
| F02F2H | Flash memory CRC operation result register | PGCRCL | R/W | – | – | √ | 0000H | F02F2H |
| F02FAH | CRC data register | CRCD | R/W | – | – | √ | 0000H | F02FAH |

**Remark** For SFRs in the SFR area, see **Table 3-5 SFR List**.

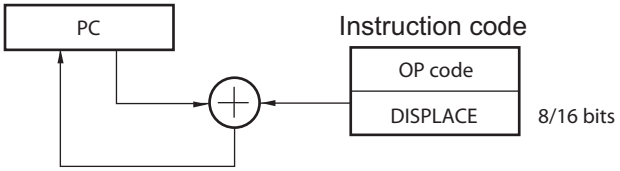
## 3.3 Instruction Address Addressing

### 3.3.1 Relative addressing

**[Function]**

Relative addressing stores in the program counter (PC) the result of adding a displacement value included in the instruction word (signed complement data: −128 to +127 or −32768 to +32767) to the program counter (PC)'s value (the start address of the next instruction), and specifies the program address to be used as the branch destination. Relative addressing is applied only to branch instructions.

**Figure 3-13. Outline of Relative Addressing**




### 3.3.2 Immediate addressing

**[Function]**

Immediate addressing stores immediate data of the instruction word in the program counter, and specifies the program address to be used as the branch destination.

For immediate addressing, CALL !!addr20 or BR !!addr20 is used to specify 20-bit addresses and CALL !addr16 or BR !addr16 is used to specify 16-bit addresses. 0000 is set to the higher 4 bits when specifying 16-bit addresses.

**Figure 3-14. Example of CALL !!addr20/BR !!addr20**

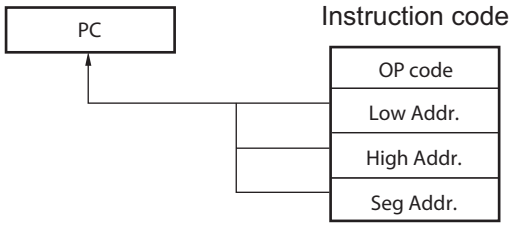
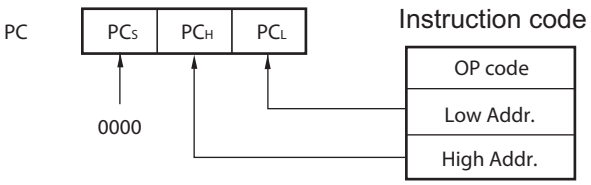



**Figure 3-15. Example of CALL !addr16/BR !addr16**



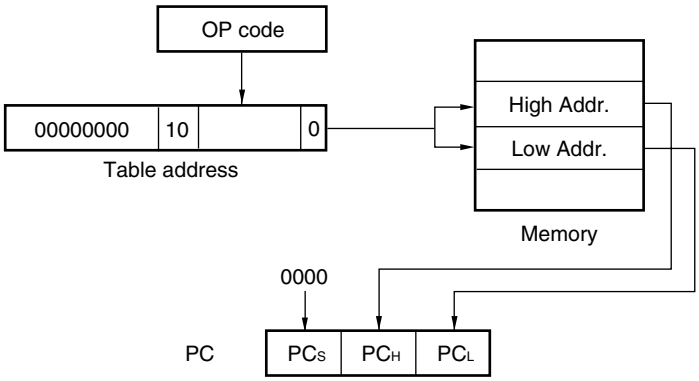
### 3.3.3 Table indirect addressing

**[Function]**

Table indirect addressing specifies a table address in the CALLT table area (0080H to 00BFH) with the 5-bit immediate data in the instruction word, stores the contents at that table address and the next address in the program counter (PC) as 16-bit data, and specifies the program address. Table indirect addressing is applied only for CALLT instructions.

In the RL78 microcontrollers, branching is enabled only to the 64 KB space from 00000H to 0FFFFH.

**Figure 3-16. Outline of Table Indirect Addressing**

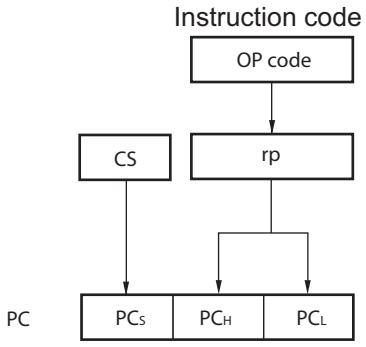


### 3.3.4 Register direct addressing

**[Function]**

Register direct addressing stores in the program counter (PC) the contents of a general-purpose register pair (AX/BC/DE/HL) and CS register of the current register bank specified with the instruction word as 20-bit data, and specifies the program address. Register direct addressing can be applied only to the CALL AX, BC, DE, HL, and BR AX instructions.

**Figure 3-17. Outline of Register Direct Addressing**



## 3.4 Addressing for Processing Data Addresses

### 3.4.1 Implied addressing

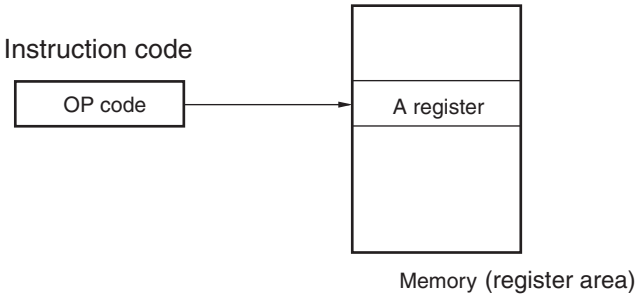
**[Function]**

Instructions for accessing registers (such as accumulators) that have special functions are directly specified with the instruction word, without using any register specification field in the instruction word.

**[Operand format]**

Implied addressing can be applied only to MULU X.

**Figure 3-18. Outline of Implied Addressing**




### 3.4.2 Register addressing

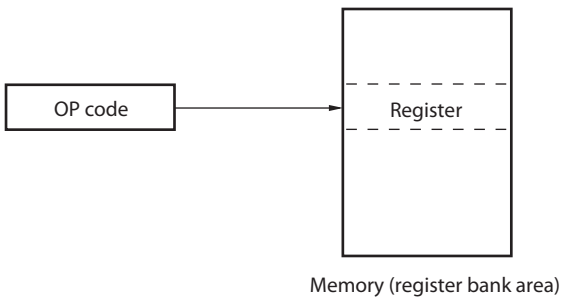
**[Function]**

Register addressing accesses a general-purpose register as an operand. The instruction word of 3-bit long is used to select an 8-bit register and the instruction word of 2-bit long is used to select a 16-bit register.

**[Operand format]**

| Identifier | Description |
|---|---|
| r | X, A, C, B, E, D, L, H |
| rp | AX, BC, DE, HL |

**Figure 3-19. Outline of Register Addressing**



### 3.4.3 Direct addressing

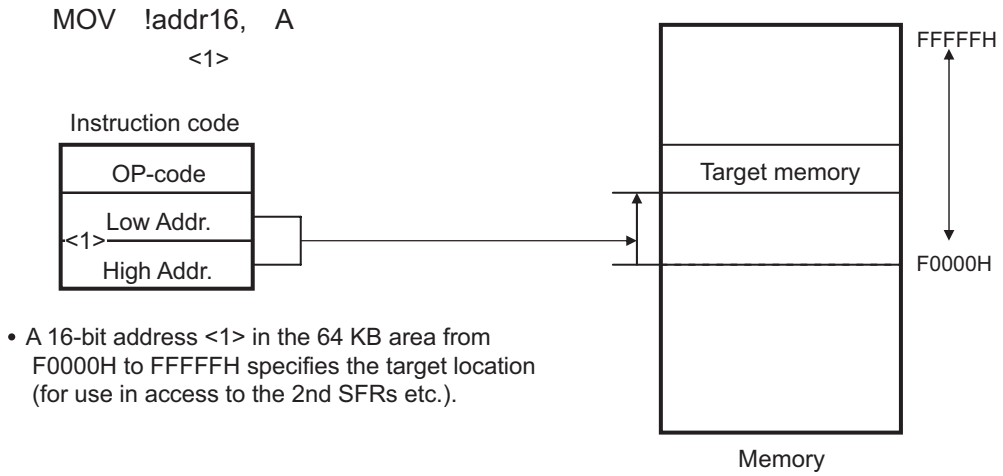
**[Function]**

Direct addressing uses immediate data in the instruction word as an operand address to directly specify the target address.

**[Operand format]**

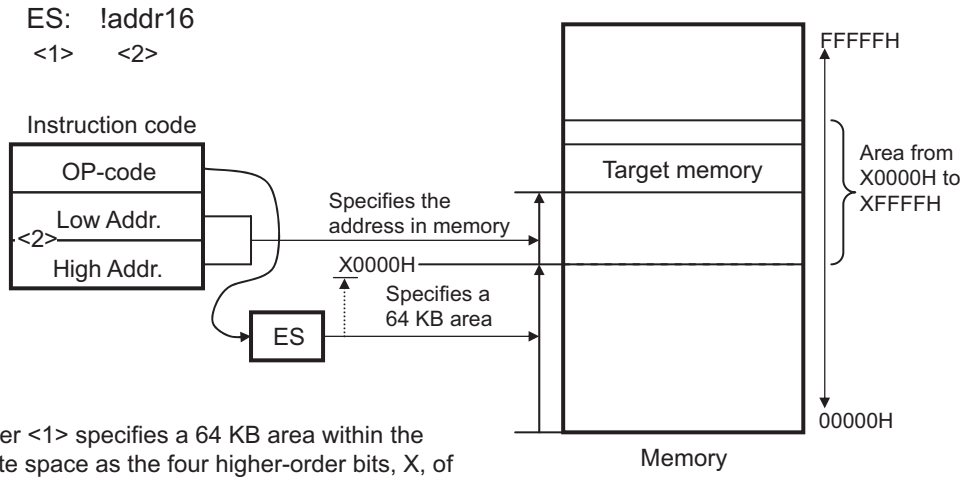
| Identifier | Description |
|---|---|
| !addr16 | Label or 16-bit immediate data (only the space from F0000H to FFFFFH is specifiable) |
| ES:!addr16 | Label or 16-bit immediate data (higher 4-bit addresses are specified by the ES register) |

**Figure 3-20. Example of !addr16**




- A 16-bit address <1> in the 64 KB area from F0000H to FFFFFH specifies the target location (for use in access to the 2nd SFRs etc.).

**Figure 3-21. Example of ES:!addr16**




- The ES register <1> specifies a 64 KB area within the overall 1-Mbyte space as the four higher-order bits, X, of the address range.
- A 16-bit address <2> in the area from X0000H to XFFFFH and the ES register <1> specify the target location; this is used for access to fixed data other than that in mirrored areas.

### 3.4.4 Short direct addressing

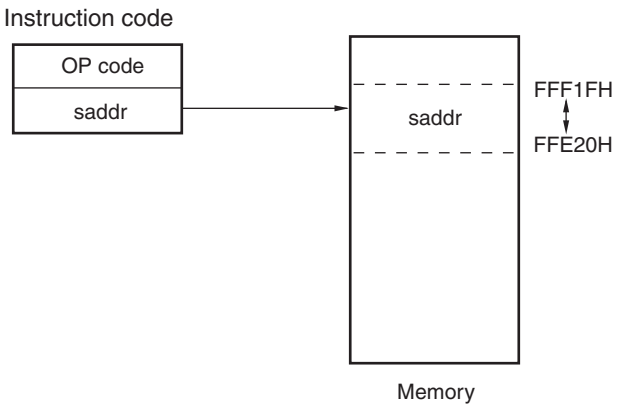
**[Function]**

Short direct addressing directly specifies the target addresses using 8-bit data in the instruction word. This type of addressing is applied only to the space from FFE20H to FFF1FH.

**[Operand format]**

| Identifier | Description |
|---|---|
| SADDR | Label, FFE20H to FFF1FH immediate data, or 0FE20H to 0FF1FH immediate data<br>(only the space from FFE20H to FFF1FH is specifiable) |
| SADDRP | Label, FFE20H to FFF1FH immediate data, or 0FE20H to 0FF1FH immediate data (even address only)<br>(only the space from FFE20H to FFF1FH is specifiable) |

**Figure 3-22. Outline of Short Direct Addressing**




**Remark** SADDR and SADDRP are used to describe the values of addresses FE20H to FF1FH with 16-bit immediate data (higher 4 bits of actual address are omitted), and the values of addresses FFE20H to FFF1FH with 20-bit immediate data.

Regardless of whether SADDR or SADDRP is used, addresses within the space from FFE20H to FFF1FH are specified for the memory.

### 3.4.5 SFR addressing

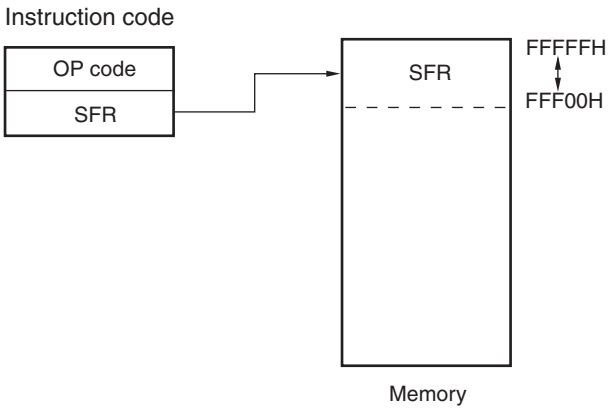
**[Function]**

SFR addressing directly specifies the target SFR addresses using 8-bit data in the instruction word. This type of addressing is applied only to the space from FFF00H to FFFFFH.

**[Operand format]**

| Identifier | Description |
|---|---|
| SFR | SFR name |
| SFRP | 16-bit-manipulatable SFR name (even address) |

**Figure 3-23. Outline of SFR Addressing**



### 3.4.6 Register indirect addressing

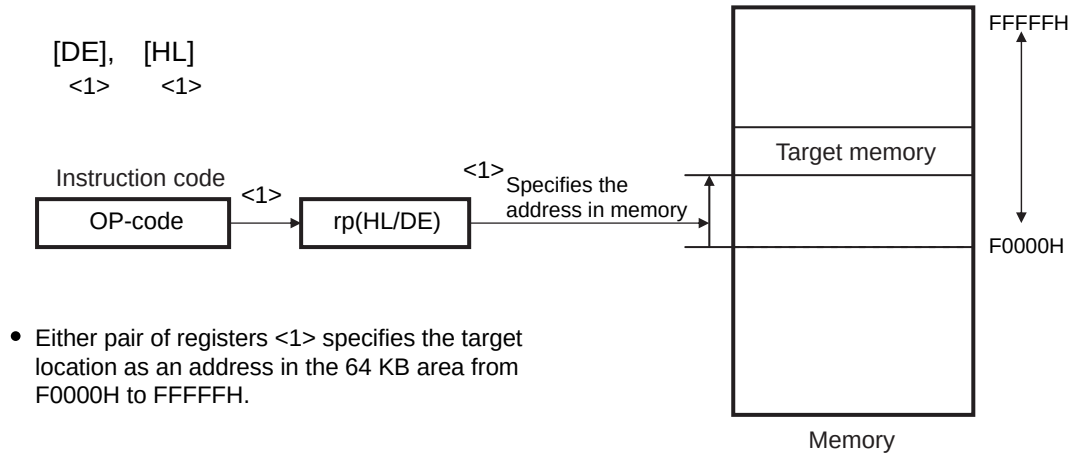
**[Function]**

Register indirect addressing directly specifies the target addresses using the contents of the register pair specified with the instruction word as an operand address.

**[Operand format]**

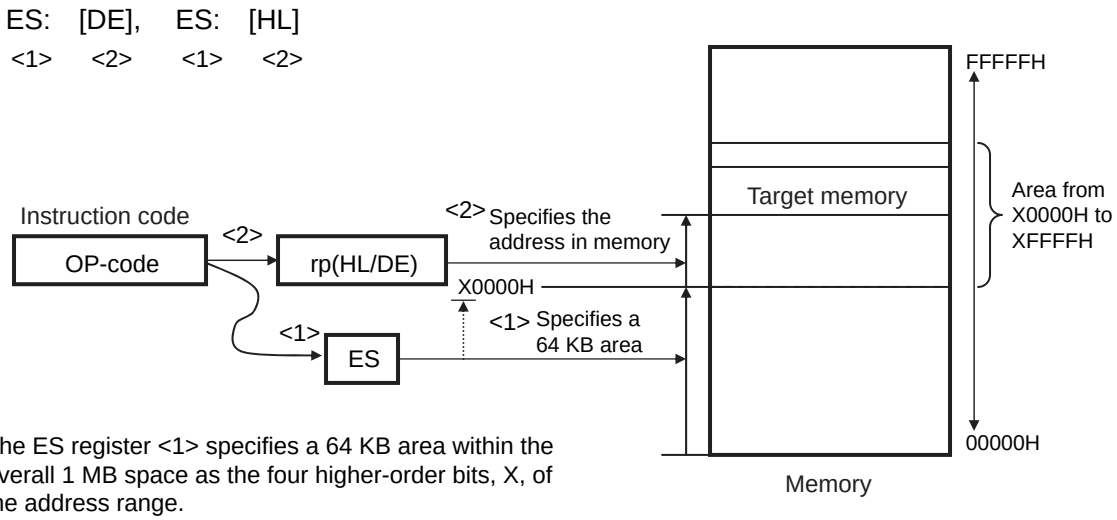
| Identifier | Description |
|---|---|
| – | [DE], [HL] (only the space from F0000H to FFFFFH is specifiable) |
| – | ES:[DE], ES:[HL] (higher 4-bit addresses are specified by the ES register) |

**Figure 3-24. Example of [DE], [HL]**




- Either pair of registers <1> specifies the target location as an address in the 64 KB area from F0000H to FFFFFH.

**Figure 3-25. Example of ES:[DE], ES:[HL]**




- The ES register <1> specifies a 64 KB area within the overall 1 MB space as the four higher-order bits, X, of the address range.
- Either pair of registers <2> and the ES register <1> specify the target location in the area from X0000H to XFFFFH.

### 3.4.7 Based addressing

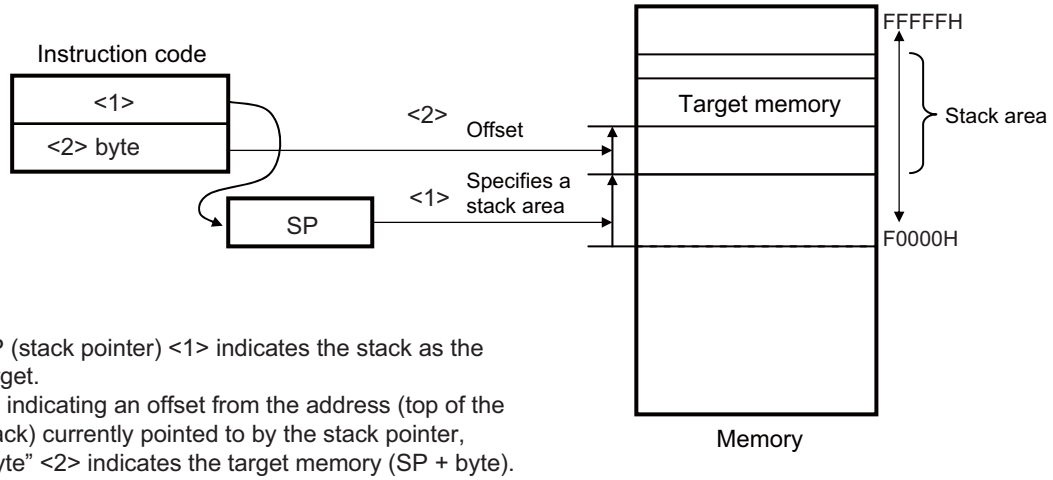
**[Function]**

Based addressing uses the contents of a register pair specified with the instruction word or 16-bit immediate data as a base address, and 8-bit immediate data or 16-bit immediate data as offset data. The sum of these values is used to specify the target address.

**[Operand format]**

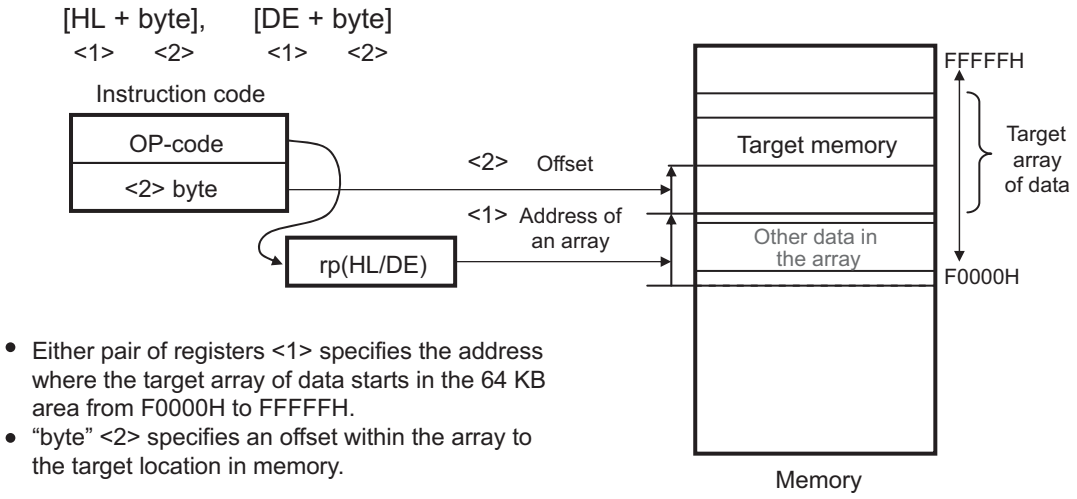
| Identifier | Description |
|---|---|
| – | [HL + byte], [DE + byte], [SP + byte] (only the space from F0000H to FFFFFH is specifiable) |
| – | word[B], word[C] (only the space from F0000H to FFFFFH is specifiable) |
| – | word[BC] (only the space from F0000H to FFFFFH is specifiable) |
| – | ES:[HL + byte], ES:[DE + byte] (higher 4-bit addresses are specified by the ES register) |
| – | ES:word[B], ES:word[C] (higher 4-bit addresses are specified by the ES register) |
| – | ES:word[BC] (higher 4-bit addresses are specified by the ES register) |

**Figure 3-26. Example of [SP+byte]**



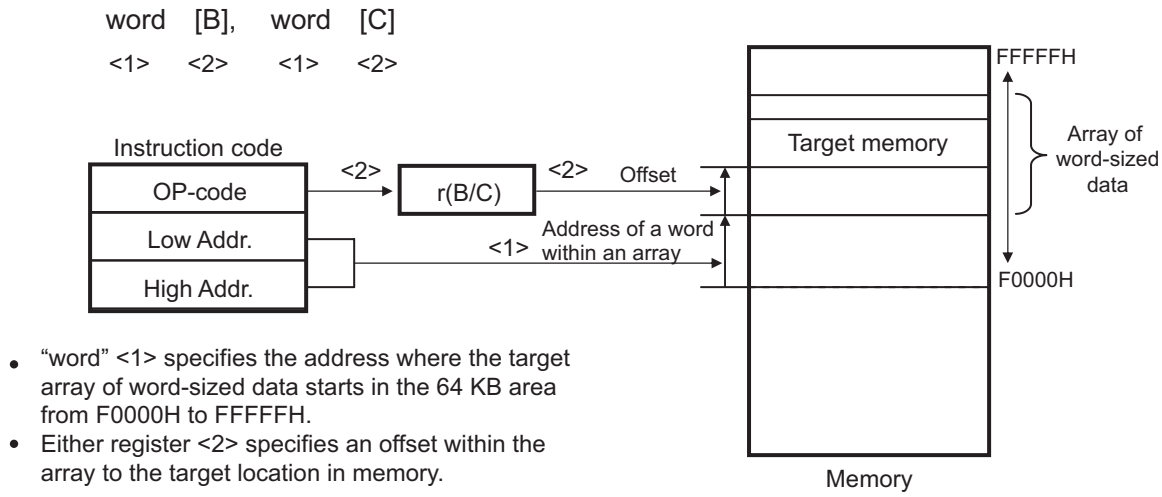

- SP (stack pointer) <1> indicates the stack as the target.
- By indicating an offset from the address (top of the stack) currently pointed to by the stack pointer, “byte” <2> indicates the target memory (SP + byte).

**Figure 3-27. Example of [HL + byte], [DE + byte]**



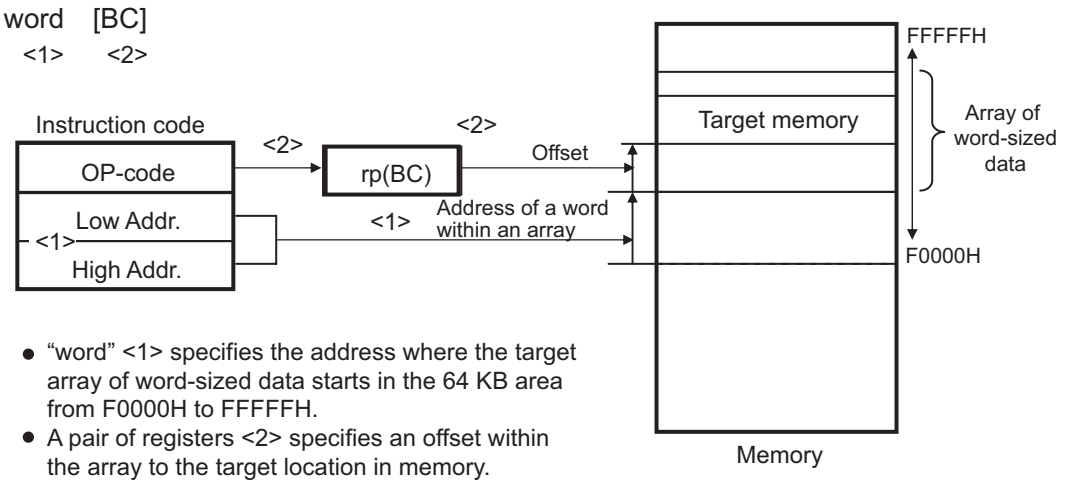

- Either pair of registers <1> specifies the address where the target array of data starts in the 64 KB area from F0000H to FFFFFH.
- "byte" <2> specifies an offset within the array to the target location in memory.

**Figure 3-28. Example of word[B], word[C]**



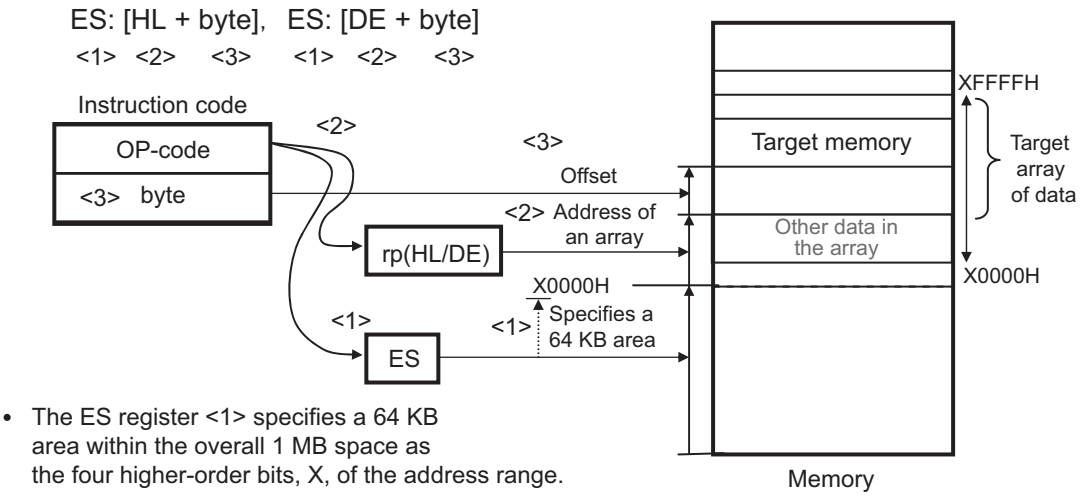

- "word" <1> specifies the address where the target array of word-sized data starts in the 64 KB area from F0000H to FFFFFH.
- Either register <2> specifies an offset within the array to the target location in memory.

**Figure 3-29. Example of word[BC]**



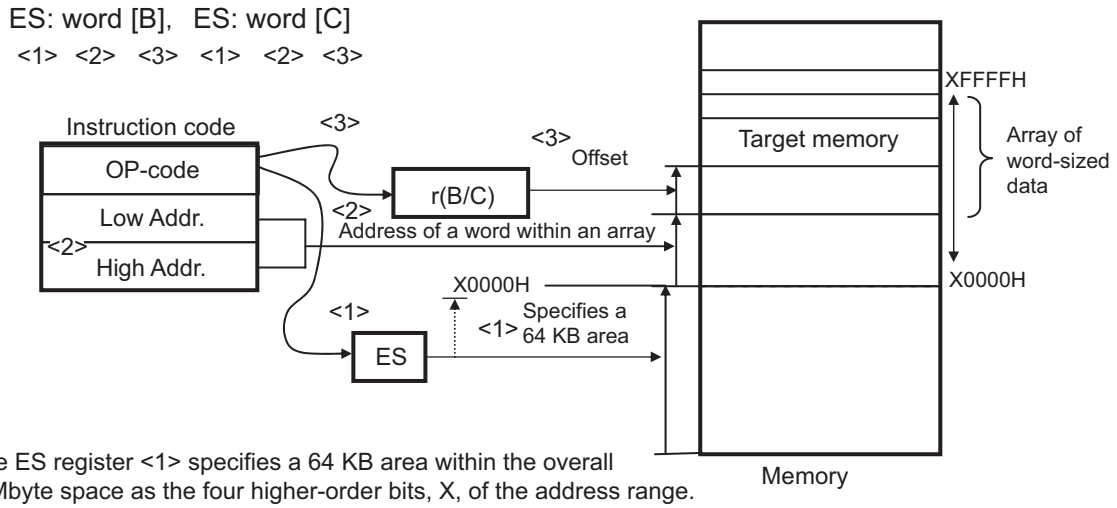

- "word" <1> specifies the address where the target array of word-sized data starts in the 64 KB area from F0000H to FFFFFH.
- A pair of registers <2> specifies an offset within the array to the target location in memory.

**Figure 3-30. Example of ES:[HL + byte], ES:[DE + byte]**



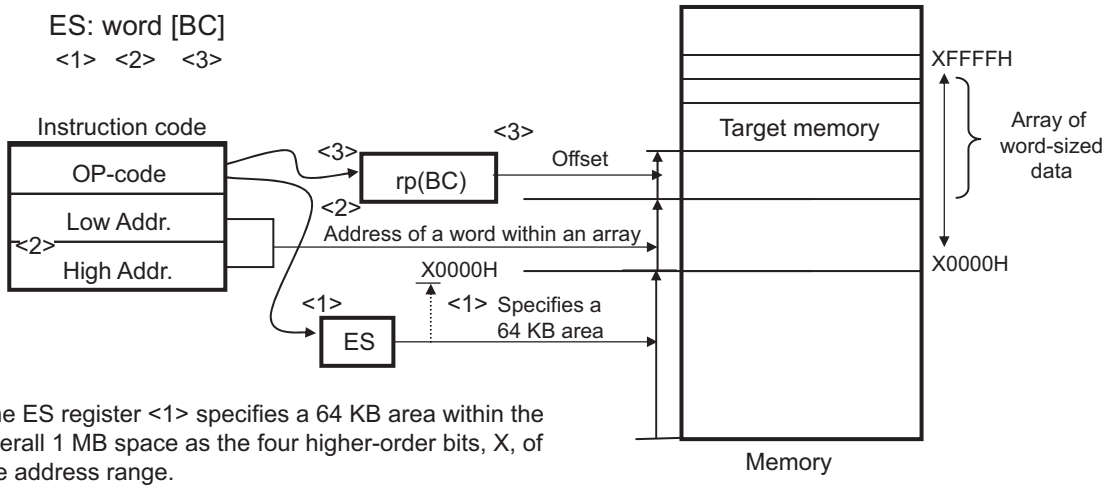

- The ES register <1> specifies a 64 KB area within the overall 1 MB space as the four higher-order bits, X, of the address range.
- Either pair of registers <2> specifies the address where the target array of data starts in the 64 KB area specified in the ES register <1>.
- "byte" <3> specifies an offset within the array to the target location in memory.

**Figure 3-31. Example of ES:word[B], ES:word[C]**




- The ES register <1> specifies a 64 KB area within the overall 1-Mbyte space as the four higher-order bits, X, of the address range.
- “word” <2> specifies the address where the target array of word-sizeddata starts in the 64 KB area specified in the ES register <1>.
- Either register <3> specifies an offset within the array tothe target location in memory.

**Figure 3-32. Example of ES:word[BC]**




- The ES register <1> specifies a 64 KB area within the overall 1 MB space as the four higher-order bits, X, of the address range.
- “word” <2> specifies the address where the target array of word-sized data starts in the 64 KB area specified in the ES register <1>.
- A pair of registers <3> specifies an offset within the array to the target location in memory.

### 3.4.8 Based indexed addressing

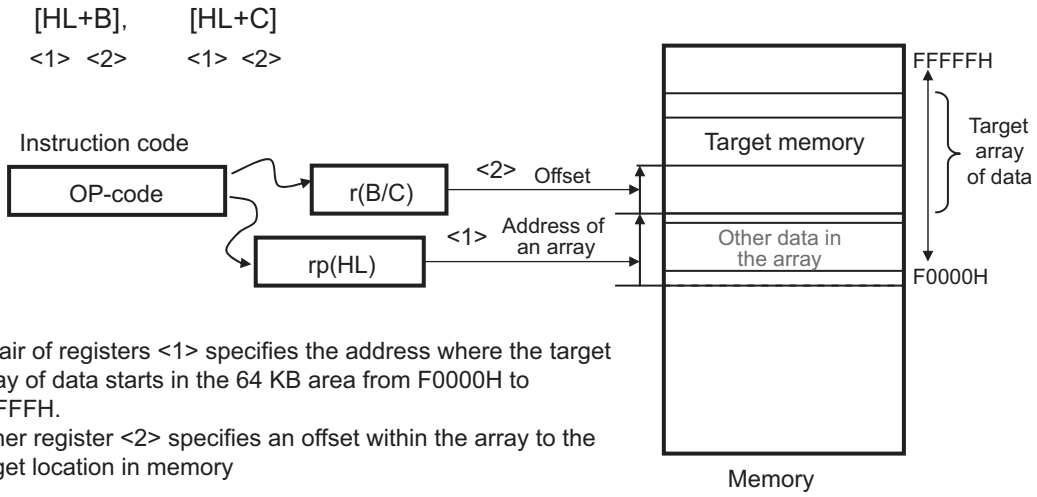
**[Function]**

Based indexed addressing uses the contents of a register pair specified with the instruction word as the base address, and the content of the B register or C register similarly specified with the instruction word as offset address. The sum of these values is used to specify the target address.

**[Operand format]**

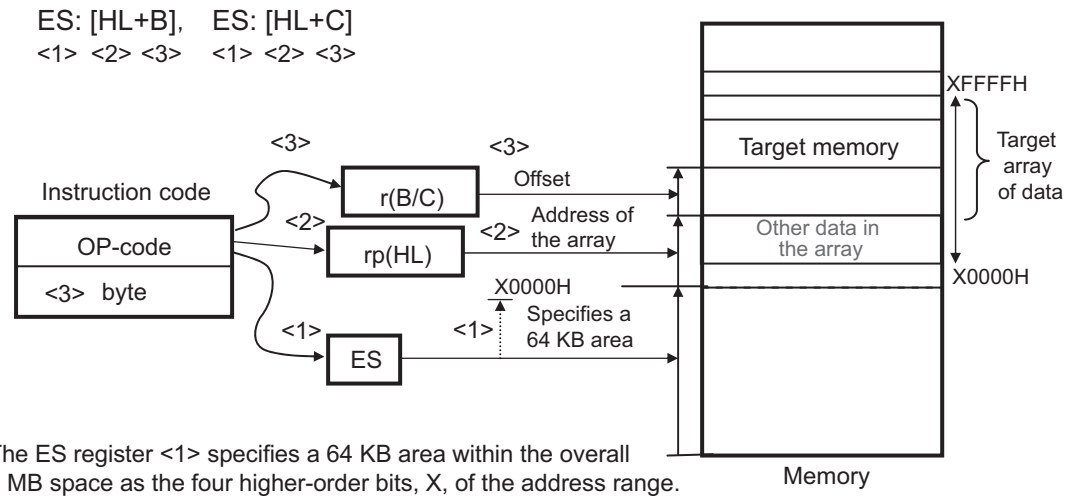
| Identifier | Description |
|---|---|
| – | [HL+B], [HL+C] (only the space from F0000H to FFFFFH is specifiable) |
| – | ES:[HL+B], ES:[HL+C] (higher 4-bit addresses are specified by the ES register) |

**Figure 3-33. Example of [HL+B], [HL+C]**




- A pair of registers <1> specifies the address where the target array of data starts in the 64 KB area from F0000H to FFFFFH.
- Either register <2> specifies an offset within the array to the target location in memory

**Figure 3-34. Example of ES:[HL+B], ES:[HL+C]**




- The ES register <1> specifies a 64 KB area within the overall 1 MB space as the four higher-order bits, X, of the address range.
- A pair of registers <2> specifies the address where the target array of data starts in the 64 KB area specified in the ES register <1>.
- Either register <3> specifies an offset within the array to the target location in memory.

### 3.4.9 Stack addressing

**[Function]**

The stack area is indirectly addressed with the stack pointer (SP) values. This addressing is automatically employed when the PUSH, POP, subroutine call, and return instructions are executed or the register is saved/restored upon generation of an interrupt request.

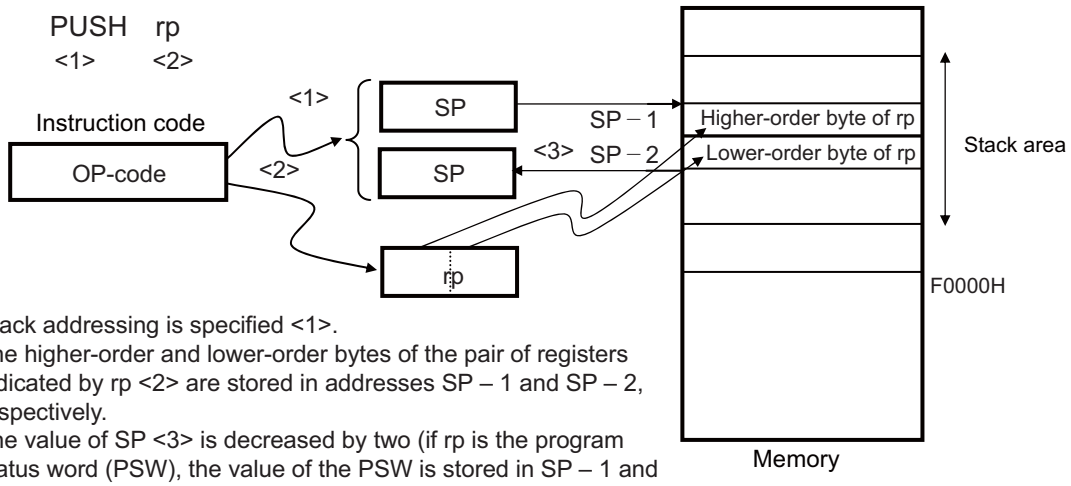
Only the internal RAM area can be set as the stack area.

**[Description format]**

| Identifier | Description |
|---|---|
| – | PUSH PSW AX/BC/DE/HL<br>POP PSW AX/BC/DE/HL<br>CALL/CALLT<br>RET<br>BRK<br>RETB<br>(Interrupt request generated)<br>RETI |

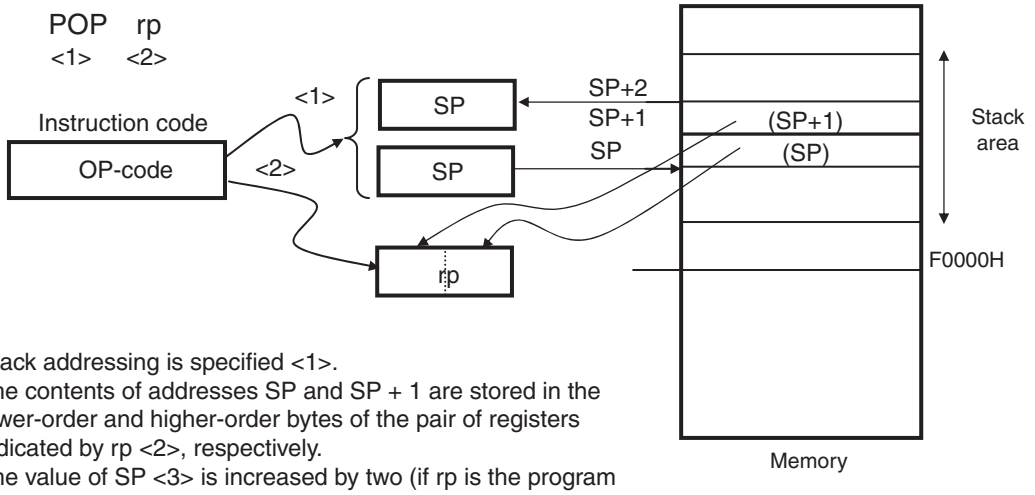
Each stack operation saves or restores data as shown in **Figures 3-35** to **3-40**.

**Figure 3-35. Example of PUSH rp**



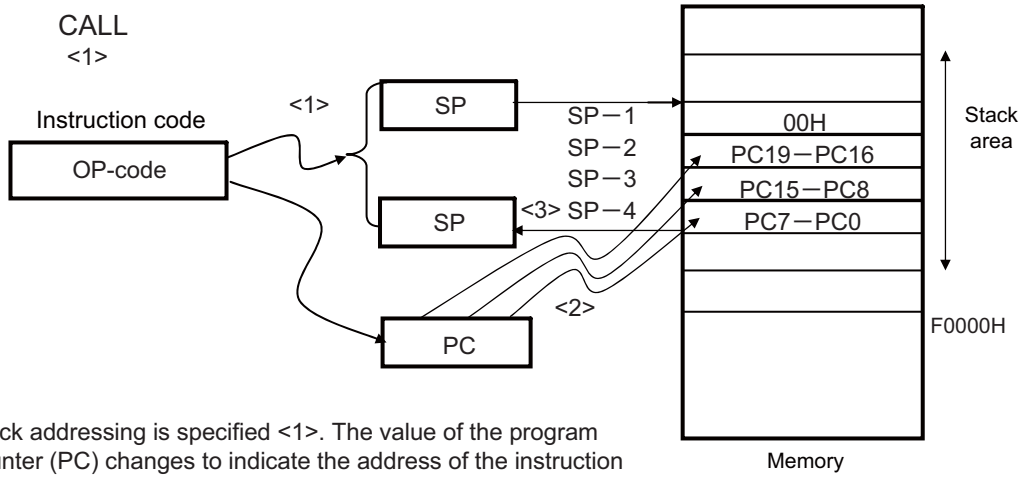

- Stack addressing is specified <1>.
- The higher-order and lower-order bytes of the pair of registers indicated by rp <2> are stored in addresses SP – 1 and SP – 2, respectively.
- The value of SP <3> is decreased by two (if rp is the program status word (PSW), the value of the PSW is stored in SP – 1 and 0 is stored in SP – 2).

**Figure 3-36. Example of POP**



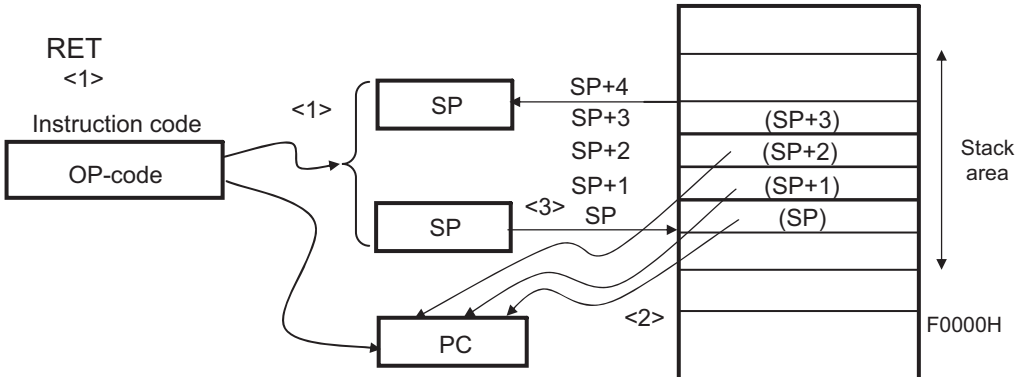

- Stack addressing is specified <1>.
- The contents of addresses SP and SP + 1 are stored in the lower-order and higher-order bytes of the pair of registers indicated by rp <2>, respectively.
- The value of SP <3> is increased by two (if rp is the program status word (PSW), the content of address SP + 1 is stored in the PSW).

**Figure 3-37. Example of CALL, CALLT**



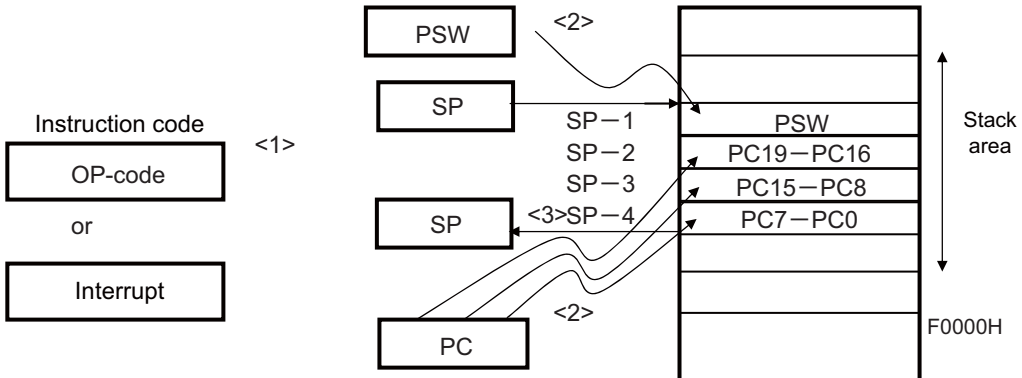

- Stack addressing is specified <1>. The value of the program counter (PC) changes to indicate the address of the instruction following the CALL instruction.
- 00H, the values of PC bits 19 to 16, 15 to 8, and 7 to 0 are stored in addresses SP – 1, SP – 2, SP – 3, and SP – 4, respectively <2>.
- The value of the SP <3> is decreased by 4.

**Figure 3-38. Example of RET**



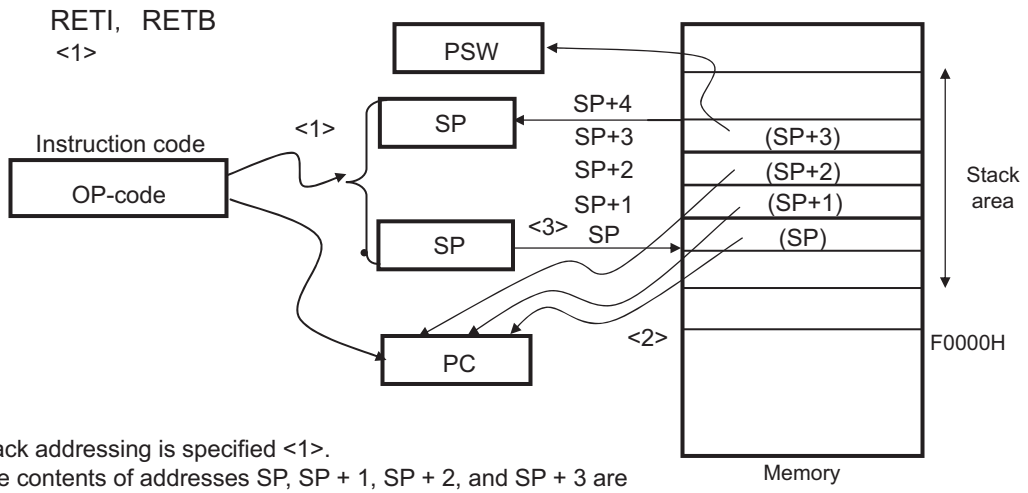

- Stack addressing is specified <1>.
- The contents of addresses SP, SP + 1, and SP + 2 are stored in PC bits 7 to 0, 15 to 8, and 19 to 16, respectively <2>.
- The value of SP <3> is increased by four.

**Figure 3-39. Example of Interrupt, BRK**




- Stack addressing is specified <1>. In response to a BRK instruction or acceptance of an interrupt, the value of the program counter (PC) changes to indicate the address of the next instruction.
- The values of the PSW, PC bits 19 to 16, 15 to 8, and 7 to 0 are stored in addresses SP – 1, SP – 2, SP – 3, and SP – 4, respectively <2>.
- The value of the SP <3> is decreased by 4.

**Figure 3-40. Example of RETI, RETB**




- Stack addressing is specified <1>.
- The contents of addresses SP, SP + 1, SP + 2, and SP + 3 are stored in PC bits 7 to 0, 15 to 8, 19 to 16, and the PSW, respectively <2>.
- The value of SP <3> is increased by four.

# CHAPTER 4 PORT FUNCTIONS

## 4.1 Port Functions

The RL78/G1A microcontrollers are provided with digital I/O ports, which enable variety of control operations.

In addition to the function as digital I/O ports, these ports have several alternate functions. For details of the alternate functions, see **CHAPTER 2 PIN FUNCTIONS**.

## 4.2 Port Configuration

Ports include the following hardware.

**Table 4-1. Port Configuration**

| Item | Configuration |
|---|---|
| Control registers | Port mode registers (PM0 to PM7, PM12, PM14, PM15)<br>Port registers (P0 to P7, P12 to P15)<br>Pull-up resistor option registers (PU0, PU1, PU3 to PU5, PU7, PU12, PU14)<br>Port input mode registers (PIM0, PIM1)<br>Port output mode registers (POM0, POM1, POM5, POM7)<br>Port mode control registers (PMC0, PMC1, PMC3 to PMC5, PMC7, PMC12)<br>A/D port configuration register (ADPC)<br>Peripheral I/O redirection register (PIOR)<br>Global digital input disable register (GDIDIS)<br>Global analog input disable register (GAIDIS) |
| Port | • 25-pin products<br><R> Total: 19 (CMOS I/O: 14 (N-ch open drain I/O [$V_{DD}$ tolerance]: 6), CMOS input: 3, N-ch open drain I/O [6-V tolerance]: 2)<br>• 32-pin products<br><R> Total: 26 (CMOS I/O: 20 (N-ch open drain I/O [$V_{DD}$ tolerance]: 9), CMOS input: 3, N-ch open drain I/O[6-V tolerance]: 3)<br>• 48-pin products<br><R> Total: 42 (CMOS I/O: 32 (N-ch open drain I/O [$V_{DD}$ tolerance]: 11), CMOS input: 5, CMOS output: 1, N-ch open drain I/O[6-V tolerance]: 4)<br>• 64-pin products<br><R> Total: 56 (CMOS I/O: 46 (N-ch open drain I/O [$V_{DD}$ tolerance]: 12), CMOS input: 5, CMOS output: 1, N-ch open drain I/O[6-V tolerance]: 4) |
| Pull-up resistor | • 25-pin products Total: 10<br>• 32-pin products Total: 15<br>• 48-pin products Total: 23<br>• 64-pin products Total: 33 |

### 4.2.1 Port 0

Port 0 is an I/O port with an output latch. Port 0 can be set to the input mode or output mode in 1-bit units using port mode register 0 (PM0). When the P00 to P06 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 0 (PU0).

Input to the P00, P01, P03 and P04 pins can be specified through a normal input buffer or a TTL input buffer in 1-bit units using port input mode register 0 (PIM0).

Output from the P02 to P04 pins can be specified as N-ch open-drain output ($V_{DD}$ tolerance[Note 1]/$EV_{DD}$ tolerance[Note 2]) in 1-bit units using port output mode register 0 (POM0).

<R> The P02 and P03 pins can be specified as digital input/output or analog input in 1-bit units, using port mode control register 0 (PMC0).

This port can also be used for timer I/O, A/D converter analog input, serial interface data I/O, clock I/O, and key interrupt input.

When reset signal is generated, the following configuration will be set.

- P00, P01 and P04 to P06 pins ··· Input mode
- P02 and P03 pins ··· Analog input

**Notes 1.** For 25- to 48-pin products

**2.** For 64-pin products

### 4.2.2 Port 1

Port 1 is an I/O port with an output latch. Port 1 can be set to the input mode or output mode in 1-bit units using port mode register 1 (PM1). When the P10 to P16 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 1 (PU1).

Input to the P10, P11, and P14 to P16 pins can be specified through a normal input buffer or a TTL input buffer in 1-bit units using port input mode register 1 (PIM1).

Output from the P10 to P15 pins can be specified as N-ch open-drain output ($V_{DD}$ tolerance[Note 1]/$EV_{DD}$ tolerance[Note 2]) in 1-bit units using port output mode register 1 (POM1).

<R> The P10 to P15 pins can be specified as digital input/output or analog input in 1-bit units, using port mode control register 1 (PMC1).

This port can also be used for A/D converter analog input, serial interface data I/O, clock I/O, programming UART I/O, timer I/O, and external interrupt request input.

When reset signal is generated, the following configuration will be set.

- P10 to P15 pins ··· Analog input
- P16 pin ··· Input mode

**Notes 1.** For 25- to 48-pin products

**2.** For 64-pin products

### 4.2.3 Port 2

Port 2 is an I/O port with an output latch. Port 2 can be set to the input mode or output mode in 1-bit units using port mode register 2 (PM2).

This port can also be used for A/D converter analog input and reference voltage input (+ side and − side).

<R> To use P20/ANI0 to P27/ANI7 as digital I/O pins, set them in the digital I/O mode by using the A/D port configuration register (ADPC). Use these pins starting from the upper bit.

To use P20/ANI0 to P27/ANI7 as analog input pins, set them in the analog input mode by using the A/D port configuration register (ADPC) and in the input mode by using the PM2 register. Use these pins starting from the lower bit.

**Table 4-2. Setting Functions of P20/ANI0 to P27/ANI7 Pins**

| ADPC Register | PM2 Register | ADS Register | P20/ANI0 to P27/ANI7 Pins |
|---|---|---|---|
| Digital I/O selection | Input mode | − | Digital input |
| | Output mode | − | Digital output |
| Analog input selection | Input mode | Selects ANI. | Analog input (to be converted) |
| | | Does not select ANI. | Analog input (not to be converted) |
| | Output mode | Selects ANI. | Setting prohibited |
| | | Does not select ANI. | |

All P20/ANI0 to P27/ANI7 are set in the analog input mode when the reset signal is generated.

### 4.2.4 Port 3

Port 3 is an I/O port with an output latch. Port 3 can be set to the input mode or output mode in 1-bit units using port mode register 3 (PM3). When the P30, P31 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 3 (PU3).

<R> The P30 and P31 pins can be specified as digital input/output or analog input in 1-bit units, using port mode control register 3 (PMC3).

This port can also be used for A/D converter analog input, external interrupt request input, real-time clock correction clock output, serial interface data I/O, clock I/O, and timer I/O.

Reset signal generation sets P30, P31 to analog input.

### 4.2.5 Port 4

Port 4 is an I/O port with an output latch. Port 4 can be set to the input mode or output mode in 1-bit units using port mode register 4 (PM4). When the P41 to P43 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 4 (PU4).

<R> The P41 pin can be specified as digital input/output or analog input, using port mode control register 4 (PMC4).

This port can also be used for A/D converter analog input, timer I/O, data I/O for a flash memory programmer/debugger.

Reset signal generation sets port 4 to input mode.

### 4.2.6 Port 5

Port 5 is an I/O port with an output latch. Port 5 can be set to the input mode or output mode in 1-bit units using port mode register 5 (PM5). When the P50, P51 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 5 (PU5).

Output from the P50 pin can be set as N-ch open-drain output ($V_{DD}$ tolerance[Note 1]/$EV_{DD}$ tolerance[Note 2]) in 1-bit units using port output mode register 5 (POM5).

<R> The P50 and 51 pins can be specified as digital input/output or analog input in 1-bit units, using port mode control register 5 (PMC5).

This port can also be used for A/D converter analog input, external interrupt request input, serial interface data I/O.

Reset signal generation sets port 5 to input mode.

**Notes 1.** For 25- to 48-pin products

**2.** For 64-pin products

### 4.2.7 Port 6

Port 6 is an I/O port with an output latch. Port 6 can be set to the input mode or output mode in 1-bit units using port mode register 6 (PM6).

The output of the P60 to P63 pins is N-ch open-drain output (6 V tolerance).

This port can also be used for serial interface data I/O and clock I/O.

Reset signal generation sets port 6 to input mode.

### 4.2.8 Port 7

Port 7 is an I/O port with an output latch. Port 7 can be set to the input mode or output mode in 1-bit units using port mode register 7 (PM7). When used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 7 (PU7).

Output from the P71 and P74 pins can be specified as N-ch open-drain output ($V_{DD}$ tolerance[Note 1]/$EV_{DD}$ tolerance[Note 2]) in 1-bit units using port output mode register 7 (POM7).

<R> The P70 pin can be specified as digital input/output or analog input, using port mode control register 7 (PMC7).

This port can also be used for A/D converter analog input, key interrupt input, serial interface data I/O, clock I/O, and external interrupt request input.

Reset signal generation sets port 7 to input mode.

**Notes 1.** For 25- to 48-pin products

**2.** For 64-pin products

### 4.2.9 Port 12

P120 is an I/O port with an output latch. Port 12 can be set to the input mode or output mode in 1-bit units using port mode register 12 (PM12). When used as an input port, use of an on-chip pull-up resistor can be specified by pull-up resistor option register 12 (PU12).

P121 to P124 are 4-bit input only ports.

<R> The P120 pin can be specified as digital input/output or analog input, using port mode control register 12 (PMC12).

This port can also be used for A/D converter analog input, connecting resonator for main system clock, connecting resonator for subsystem clock, external clock input for main system clock, and external clock input for subsystem clock.

Reset signal generation sets P120 to analog input, and sets P121 to P124 to input mode.

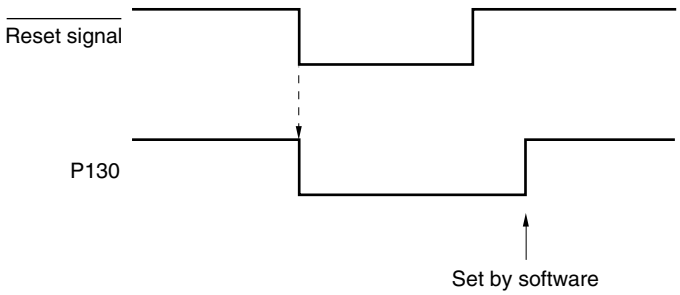
### 4.2.10 Port 13

P130 is a 1-bit output-only port with an output latch. P137 is a 1-bit input-only port.

P130 is fixed an output port, and P137 is fixed an input ports.

This port can also be used for external interrupt request input.

**Remark** When reset is effected, P130 outputs a low level. If P130 is set to output a high level before reset is effected, the output signal of P130 can be dummy-output as the CPU reset signal.



### 4.2.11 Port 14

Port 14 is an I/O port with an output latch. Port 14 can be set to the input mode or output mode in 1-bit units using port mode register 14 (PM14). When the P140, P141 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 14 (PU14).

This port can also be used for clock/buzzer output, and external interrupt request input,

Reset signal generation sets P140, P141 to input mode.

### 4.2.12 Port 15

Port 15 is an I/O port with an output latch. Port 15 can be set to the input mode or output mode in 1-bit units using port mode register 15 (PM15).

This port can also be used for A/D converter analog input.

<R> To use P150/ANI8 to P154/ANI12 as digital input pins, set them in the digital I/O mode by using the A/D port configuration register (ADPC) and in the input mode by using the PM15 register. Use these pins starting from the upper bit.

<R> To use 150/ANI8 to P154/ANI12 as digital output pins, set them in the digital I/O mode by using the A/D port configuration register (ADPC) and in the output mode by using the PM15 register. Use these pins starting from the upper bit.

To use 150/ANI8 to P154/ANI12 as analog input pins, set them in the analog input mode by using the A/D port configuration register (ADPC) and in the input mode by using the PM15 register. Use these pins starting from the lower bit.

**Table 4-3. Setting Functions of P150/ANI8 to P154/ANI12 Pins**

| ADPC Register | PM15 Register | ADS Register | P150/ANI8 to P154/ANI12 Pins |
|---|---|---|---|
| Digital I/O selection | Input mode | – | Digital input |
| | Output mode | – | Digital output |
| Analog input selection | Input mode | Selects ANI. | Analog input (to be converted) |
| | | Does not select ANI. | Analog input (not to be converted) |
| | Output mode | Selects ANI. | Setting prohibited |
| | | Does not select ANI. | |

All P150/ANI8 to P154/ANI12 are set in the analog input mode when the reset signal is generated.

## 4.3 Registers Controlling Port Function

Port functions are controlled by the following registers.

- Port mode registers (PMxx)
- Port registers (Pxx)
- Pull-up resistor option registers (PUxx)
- Port input mode registers (PIMxx)
- Port output mode registers (POMxx)
- Port mode control registers (PMCxx)
- A/D port configuration register (ADPC)
- Peripheral I/O redirection register (PIOR)
- Global digital input disable register (GDIDIS)
- Global analog input disable register (GAIDIS)

**Caution Which registers and bits are included depends on the product. For registers and bits mounted on each product, see Table 4-4 Be sure to set bits that are not mounted to their initial values.**

**Table 4-4. PMxx, Pxx, PUxx, PIMxx, POMxx, PMCxx registers and the bits mounted on each product (1/2)**

| Port | | Bit Name | | | | | | 64-pin | 48-pin | 32-pin | 25-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | PMxx Register | Pxx Register | PUxx Register | PIMxx Register | POMxx Register | PMCxx Register | | | | |
| Port 0 | 0 | PM00 | P00 | PU00 | PIM00 | – | – | √ | – | – | – |
| | 1 | PM01 | P01 | PU01 | PIM01 | – | – | √ | – | – | – |
| | 2 | PM02 | P02 | PU02 | – | POM02 | PMC02 | √ | √ | √ | √ |
| | 3 | PM03 | P03 | PU03 | PIM03 | POM03 | PMC03 | √ | √ | √ | √ |
| | 4 | PM04 | P04 | PU04 | PIM04 | POM04 | – | √ | – | – | – |
| | 5 | PM05 | P05 | PU05 | – | – | – | √ | – | – | – |
| | 6 | PM06 | P06 | PU06 | – | – | – | √ | – | – | – |
| Port 1 | 0 | PM10 | P10 | PU10 | PIM10 | POM10 | PMC00 | √ | √ | √ | √ |
| | 1 | PM11 | P11 | PU11 | PIM11 | POM11 | PMC01 | √ | √ | √ | √ |
| | 2 | PM12 | P12 | PU12 | – | POM12 | PMC02 | √ | √ | √ | √ |
| | 3 | PM13 | P13 | PU13 | – | POM13 | PMC03 | √ | √ | √ | – |
| | 4 | PM14 | P14 | PU14 | PIM14 | POM14 | PMC04 | √ | √ | √ | – |
| | 5 | PM15 | P15 | PU15 | PIM15 | POM15 | PMC05 | √ | √ | √ | – |
| | 6 | PM16 | P16 | PU16 | PIM16 | – | – | √ | √ | – | – |
| Port 2 | 0 | PM20 | P20 | – | – | – | – | √ | √ | √ | √ |
| | 1 | PM21 | P21 | – | – | – | – | √ | √ | √ | √ |
| | 2 | PM22 | P22 | – | – | – | – | √ | √ | √ | √ |
| | 3 | PM23 | P23 | – | – | – | – | √ | √ | √ | √ |
| | 4 | PM24 | P24 | – | – | – | – | √ | √ | √ | – |
| | 5 | PM25 | P25 | – | – | – | – | √ | √ | – | – |
| | 6 | PM26 | P26 | – | – | – | – | √ | √ | – | – |
| | 7 | PM27 | P27 | – | – | – | – | √ | √ | – | – |

**Remark** √: Mounted, –: Not mounted

**Table 4-4. PMxx, Pxx, PUxx, PIMxx, POMxx, PMCxx registers and the bits mounted on each product (2/2)**

| Port | | Bit Name | | | | | | 64-pin | 48-pin | 32-pin | 25-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | PMxx Register | Pxx Register | PUxx Register | PIMxx Register | POMxx Register | PMCxx Register | | | | |
| Port 3 | 0 | PM30 | P30 | PU30 | – | – | PMC30 | √ | √ | √ | √ |
| | 1 | PM31 | P31 | PU31 | – | – | PMC31 | √ | √ | √ | √ |
| Port 4 | 0 | PM40 | P40 | PU40 | – | – | – | √ | √ | √ | √ |
| | 1 | PM41 | P41 | PU41 | – | – | PMC41 | √ | √ | – | – |
| | 2 | PM42 | P42 | PU42 | – | – | – | √ | – | – | – |
| | 3 | PM43 | P43 | PU43 | – | – | – | √ | – | – | – |
| Port 5 | 0 | PM50 | P50 | PU50 | – | POM50 | PMC50 | √ | √ | √ | √ |
| | 1 | PM51 | P51 | PU51 | – | – | PMC51 | √ | √ | √ | √ |
| Port 6 | 0 | PM60 | P60 | – | – | – | – | √ | √ | √ | √ |
| | 1 | PM61 | P61 | – | – | – | – | √ | √ | √ | √ |
| | 2 | PM62 | P62 | – | – | – | – | √ | √ | √ | – |
| | 3 | PM63 | P63 | – | – | – | – | √ | √ | – | – |
| Port 7 | 0 | PM70 | P70 | PU70 | – | – | PMC70 | √ | √ | √ | – |
| | 1 | PM71 | P71 | PU71 | – | POM71 | – | √ | √ | – | – |
| | 2 | PM72 | P72 | PU72 | – | – | – | √ | √ | – | – |
| | 3 | PM73 | P73 | PU73 | – | – | – | √ | √ | – | – |
| | 4 | PM74 | P74 | PU74 | – | POM74 | – | √ | √ | – | – |
| | 5 | PM75 | P75 | PU75 | – | – | – | √ | √ | – | – |
| | 6 | PM76 | P76 | PU76 | – | – | – | √ | – | – | – |
| | 7 | PM77 | P77 | PU77 | – | – | – | √ | – | – | – |
| Port 12 | 0 | PM120 | P120 | PU120 | – | – | PMC120 | √ | √ | √ | – |
| | 1 | – | P121 | – | – | – | – | √ | √ | √ | √ |
| | 2 | – | P122 | – | – | – | – | √ | √ | √ | √ |
| | 3 | – | P123 | – | – | – | – | √ | √ | – | – |
| | 4 | – | P124 | – | – | – | – | √ | √ | – | – |
| Port 13 | 0 | – | P130 | – | – | – | – | √ | √ | – | – |
| | 7 | – | P137 | – | – | – | – | √ | √ | √ | √ |
| Port 14 | 0 | PM140 | P140 | PU140 | – | – | – | √ | √ | – | – |
| | 1 | PM141 | P141 | PU141 | – | – | – | √ | – | – | – |
| Port 15 | – | PM150 | P150 | – | – | – | – | √ | √ | – | – |
| | – | PM151 | P151 | – | – | – | – | √ | – | – | – |
| | – | PM152 | P152 | – | – | – | – | √ | – | – | – |
| | – | PM153 | P153 | – | – | – | – | √ | – | – | – |
| | – | PM154 | P154 | – | – | – | – | √ | – | – | – |

**Remark** √: Mounted, –: Not mounted

### 4.3.1 Port mode registers (PMxx)

These registers specify input or output mode for the port in 1-bit units.

These registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets these registers to FFH.

<R> When port pins are used as alternate-function pins, set the port mode register by referencing **4.5 Register Settings When Using Alternate Function**.

**Figure 4-1. Format of Port Mode Register**

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset | R/W |
|---|---|---|---|---|---|---|---|---|---|---|---|
| PM0 | 1 | PM06 | PM05 | PM04 | PM03 | PM02 | PM01 | PM00 | FFF20H | FFH | R/W |
| PM1 | 1 | PM16 | PM15 | PM14 | PM13 | PM12 | PM11 | PM10 | FFF21H | FFH | R/W |
| PM2 | PM27 | PM26 | PM25 | PM24 | PM23 | PM22 | PM21 | PM20 | FFF22H | FFH | R/W |
| PM3 | 1 | 1 | 1 | 1 | 1 | 1 | PM31 | PM30 | FFF23H | FFH | R/W |
| PM4 | 1 | 1 | 1 | 1 | PM43 | PM42 | PM41 | PM40 | FFF24H | FFH | R/W |
| PM5 | 1 | 1 | 1 | 1 | 1 | 1 | PM51 | PM50 | FFF25H | FFH | R/W |
| PM6 | 1 | 1 | 1 | 1 | PM63 | PM62 | PM61 | PM60 | FFF26H | FFH | R/W |
| PM7 | PM77 | PM76 | PM75 | PM74 | PM73 | PM72 | PM71 | PM70 | FFF27H | FFH | R/W |
| PM12 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | PM120 | FFF2CH | FFH | R/W |
| PM14 | 1 | 1 | 1 | 1 | 1 | 1 | PM141 | PM140 | FFF2EH | FFH | R/W |
| PM15 | 1 | 1 | 1 | PM154 | PM153 | PM152 | PM151 | PM150 | FFF2FH | FFH | R/W |

| PMmn | Pmn pin I/O mode selection<br>(m = 0 to 7, 12, 14, 15; n = 0 to 7) |
|---|---|
| 0 | Output mode (output buffer on) |
| 1 | Input mode (output buffer off) |

<R> **Caution Be sure to set bits that are not mounted to their initial values.**

### 4.3.2 Port registers (Pxx)

These registers set the output latch value of a port.

If the data is read in the input mode, the pin level is read. If it is read in the output mode, the output latch value is read[Note].

These registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H.

**Note** If P02, P03, P10 to P15, P20 to P27, P30, P31, P41, P50, P51, P70, P120, and P150 to P154 are set up as analog inputs of the A/D converter, when a port is read while in the input mode, 0 is always returned, not the pin level.

**Figure 4-2. Format of Port Register**

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset | R/W |
|---|---|---|---|---|---|---|---|---|---|---|---|
| P0 | 0 | P06 | P05 | P04 | P03 | P02 | P01 | P00 | FFF00H | 00H (output latch) | R/W |
| P1 | 0 | P16 | P15 | P14 | P13 | P12 | P11 | P10 | FFF01H | 00H (output latch) | R/W |
| P2 | P27 | P26 | P25 | P24 | P23 | P22 | P21 | P20 | FFF02H | 00H (output latch) | R/W |
| P3 | 0 | 0 | 0 | 0 | 0 | 0 | P31 | P30 | FFF03H | 00H (output latch) | R/W |
| P4 | 0 | 0 | 0 | 0 | P43 | P42 | P41 | P40 | FFF04H | 00H (output latch) | R/W |
| P5 | 0 | 0 | 0 | 0 | 0 | 0 | P51 | P50 | FFF05H | 00H (output latch) | R/W |
| P6 | 0 | 0 | 0 | 0 | P63 | P62 | P61 | P60 | FFF06H | 00H (output latch) | R/W |
| P7 | P77 | P76 | P75 | P74 | P73 | P72 | P71 | P70 | FFF07H | 00H (output latch) | R/W |
| P12 | 0 | 0 | 0 | P124 | P123 | P122 | P121 | P120 | FFF0CH | Undefined | R/W[Note 1] |
| P13 | P137 | 0 | 0 | 0 | 0 | 0 | 0 | P130 | FFF0DH | **Note 2** | R/W[Note 1] |
| P14 | 0 | 0 | 0 | 0 | 0 | 0 | P141 | P140 | FFF0EH | 00H (output latch) | R/W |
| P15 | 0 | 0 | 0 | P154 | P153 | P152 | P151 | P150 | FFF0FH | 00H (output latch) | R/W |

| Pmn | Output data control (in output mode) | Input data read (in input mode) |
|---|---|---|
| 0 | Output 0 | Input low level |
| 1 | Output 1 | Input high level |

**Notes 1.** P121 to P124, and P137 are read-only.

**2.** P137 : Undefined
P1301: 0 (output latch)

<R> **Caution Be sure to set bits that are not mounted to their initial values.**

**Remark** m = 0 to 7, 12 to 15; n = 0 to 7

### 4.3.3 Pull-up resistor option registers (PUxx)

These registers specify whether the on-chip pull-up resistors are to be used or not. On-chip pull-up resistors can be <R> used in 1-bit units only for the bits set to both normal output mode (POMmn = 0) and input mode (PMmn = 1) for the pins to which the use of an on-chip pull-up resistor has been specified in these registers. On-chip pull-up resistors cannot be connected to bits set to output mode and bits used as alternate-function output pins and analog setting (PMC = 1, ADPC = 1), regardless of the settings of these registers.

These registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H (Only PU4 is set to 01H).

**Caution When a port with the PIMn register is input from different potential device to the TTL buffer, pull up to the power supply of the different potential device via an external resistor by setting PUmn = 0.**

**Figure 4-3. Format of Pull-up Resistor Option Register**

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset | R/W |
|---|---|---|---|---|---|---|---|---|---|---|---|
| PU0 | 0 | PU06 | PU05 | PU04 | PU03 | PU02 | PU01 | PU00 | F0030H | 00H | R/W |
| PU1 | 0 | PU16 | PU15 | PU14 | PU13 | PU12 | PU11 | PU10 | F0031H | 00H | R/W |
| PU3 | 0 | 0 | 0 | 0 | 0 | 0 | PU31 | PU30 | F0033H | 00H | R/W |
| PU4 | 0 | 0 | 0 | 0 | PU43 | PU42 | PU41 | PU40 | F0034H | 01H | R/W |
| PU5 | 0 | 0 | 0 | 0 | 0 | 0 | PU51 | PU50 | F0035H | 00H | R/W |
| PU7 | PU77 | PU76 | PU75 | PU74 | PU73 | PU72 | PU71 | PU70 | F0037H | 00H | R/W |
| PU12 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PU120 | F003CH | 00H | R/W |
| PU14 | 0 | 0 | 0 | 0 | 0 | 0 | PU141 | PU140 | F003EH | 00H | R/W |

| PUmn | Pmn pin on-chip pull-up resistor selection<br>(m = 0, 1, 3 to 5 , 7, 12, 14; n = 0 to 7) |
|---|---|
| 0 | On-chip pull-up resistor not connected |
| 1 | On-chip pull-up resistor connected |

<R> **Caution Be sure to set bits that are not mounted to their initial values.**

### 4.3.4 Port input mode registers (PIMxx)

These registers set the input buffer in 1-bit units.

TTL input buffer can be selected during serial communication with an external device of the different potential.

Port input mode registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H.

**Figure 4-4. Format of Port Input Mode Register**

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset | R/W |
|---|---|---|---|---|---|---|---|---|---|---|---|
| PIM0 | 0 | 0 | 0 | PIM04 | PIM03 | 0 | PIM01 | PIM00 | F0040H | 00H | R/W |
| PIM1 | 0 | PIM16 | PIM15 | PIM14 | 0 | 0 | PIM11 | PIM10 | F0041H | 00H | R/W |

| PIMmn | Pmn pin input buffer selection (m = 0, 1 ; n = 0, 1, 3 to 6) |
|---|---|
| 0 | Normal input buffer |
| 1 | TTL input buffer |

<R> **Caution Be sure to set bits that are not mounted to their initial values.**

### 4.3.5 Port output mode registers (POMxx)

These registers set the output mode in 1-bit units.

N-ch open drain output ($V_{DD}$ tolerance[Note 1]/$EV_{DD}$ tolerance[Note 2]) mode can be selected during serial communication with an external device of the different potential, and for the SDA00, SDA01, SDA10, SDA11, SDA20, and SDA21 pins during simplified $I^2C$ communication with an external device of the same potential.

In addition, POMxx register is set with PUxx register, whether or not to use the on-chip pull-up resistor.

These registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H.

<R> **Caution An on-chip pull-up resistor is not connected to a bit for which N-ch open drain output ($V_{DD}$ tolerance[Note 1]/$EV_{DD}$ tolerance[Note 2]) mode (POMmn = 1) is set.**

**Figure 4-5. Format of Port Input Mode Register**

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset | R/W |
|---|---|---|---|---|---|---|---|---|---|---|---|
| POM0 | 0 | 0 | 0 | POM04 | POM03 | POM02 | 0 | 0 | F0050H | 00H | R/W |
| POM1 | 0 | 0 | POM15 | POM14 | POM13 | POM12 | POM11 | POM10 | F0051H | 00H | R/W |
| POM5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | POM50 | F0055H | 00H | R/W |
| POM7 | 0 | 0 | 0 | POM74 | 0 | 0 | POM71 | 0 | F0057H | 00H | R/W |

| | POMmn | Pmn pin output mode selection (m = 0, 1, 5, 7; n = 0 to 5) |
|---|---|---|
| <R> | 0 | Normal output mode |
| <R> | 1 | N-ch open-drain output ($V_{DD}$ tolerance[Note 1]/$EV_{DD}$ tolerance[Note 2]) mode |

**Notes 1.** For 25- to 48-pin products

**2.** For 64-pin products

<R> **Caution Be sure to set bits that are not mounted to their initial values.**

### 4.3.6 Port mode control registers (PMCxx)

These registers set the digital I/O/analog input in 1-bit units.

Port mode control registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to FFH.

**Figure 4-6. Format of Port Mode Control Register**

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset | R/W |
|---|---|---|---|---|---|---|---|---|---|---|---|
| PMC0 | 1 | 1 | 1 | 1 | PMC03 | PMC02 | 1 | 1 | F0060H | FFH | R/W |
| PMC1 | 1 | 1 | PMC15 | PMC14 | PMC13 | PMC12 | PMC11 | PMC10 | F0061H | FFH | R/W |
| PMC3 | 1 | 1 | 1 | 1 | 1 | 1 | PMC31 | PMC30 | F0063H | FFH | R/W |
| PMC4 | 1 | 1 | 1 | 1 | 1 | 1 | PMC41 | 1 | F0064H | FFH | R/W |
| PMC5 | 1 | 1 | 1 | 1 | 1 | 1 | PMC51 | PMC50 | F0065H | FFH | R/W |
| PMC7 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | PMC70 | F0067H | FFH | R/W |
| PMC12 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | PMC120 | F006CH | FFH | R/W |

| PMCmn | Pmn pin digital I/O/analog input selection<br>(m = 0, 1, 3 to 5, 7, 12; n = 0 to 5) |
|---|---|
| 0 | Digital I/O (alternate function other than analog input) |
| 1 | Analog input |

<R> **Cautions 1. Select input mode by using port mode registers 0, 1, 3 to 5, 7, 12 (PM0, PM1, PM3 to PM5, PM7, PM12) for the ports which are set by the PMCxx register as analog input.**

**2. Do not set the pin set by the PMCxx register as digital I/O by the analog input channel specification register (ADS).**

<R> **3. Be sure to set bits that are not mounted to their initial values.**

### 4.3.7 A/D port configuration register (ADPC)

This register switches the P20/ANI0 to P27/ANI7, and P150/ANI8 to P154/ANI12 pins to digital I/O of port or analog input of A/D converter.

The ADPC register can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

**Figure 4-7. Format of A/D Port Configuration Register (ADPC)**

Address: F0076H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADPC | 0 | 0 | 0 | 0 | ADPC3 | ADPC2 | ADPC1 | ADPC0 |

| ADPC3 | ADPC2 | ADPC1 | ADPC0 | Analog input (A)/digital I/O (D) switching | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | ANI12/P154 | ANI11/P153 | ANI10/P152 | ANI9/P151 | ANI8/P150 | ANI7/P27 | ANI6/P26 | ANI5/P25 | ANI4/P24 | ANI3/P23 | ANI2/P22 | ANI1/P21 | ANI0/P20 |
| 0 | 0 | 0 | 0 | A | A | A | A | A | A | A | A | A | A | A | A | A |
| 0 | 0 | 0 | 1 | D | D | D | D | D | D | D | D | D | D | D | D | D |
| 0 | 0 | 1 | 0 | D | D | D | D | D | D | D | D | D | D | D | D | A |
| 0 | 0 | 1 | 1 | D | D | D | D | D | D | D | D | D | D | D | A | A |
| 0 | 1 | 0 | 0 | D | D | D | D | D | D | D | D | D | D | A | A | A |
| 0 | 1 | 0 | 1 | D | D | D | D | D | D | D | D | D | A | A | A | A |
| 0 | 1 | 1 | 0 | D | D | D | D | D | D | D | D | A | A | A | A | A |
| 0 | 1 | 1 | 1 | D | D | D | D | D | D | D | A | A | A | A | A | A |
| 1 | 0 | 0 | 0 | D | D | D | D | D | D | A | A | A | A | A | A | A |
| 1 | 0 | 0 | 1 | D | D | D | D | D | A | A | A | A | A | A | A | A |
| 1 | 0 | 1 | 0 | D | D | D | D | A | A | A | A | A | A | A | A | A |
| 1 | 0 | 1 | 1 | D | D | D | A | A | A | A | A | A | A | A | A | A |
| 1 | 1 | 0 | 0 | D | D | A | A | A | A | A | A | A | A | A | A | A |
| 1 | 1 | 0 | 1 | D | A | A | A | A | A | A | A | A | A | A | A | A |
| 1 | 1 | 1 | 0 | A | A | A | A | A | A | A | A | A | A | A | A | A |
| 1 | 1 | 1 | 1 | A | A | A | A | A | A | A | A | A | A | A | A | A |

**Cautions 1. Set the port to analog input by ADPC register to the input mode by using port mode registers 2, 15 (PM2, PM15).**

**2. Do not set the pin set by the ADPC register as digital I/O by the analog input channel specification register (ADS).**

<R> **3. When using $AV_{REFP}$ and $AV_{REFM}$, set ANI0 and ANI1 to analog input and set the port mode register to the input mode.**

### 4.3.8 Peripheral I/O redirection register (PIOR)

This register is used to specify whether to enable or disable the peripheral I/O redirect function.

This function is used to switch ports to which alternate functions are assigned.

Use the PIOR register to assign a port to the function to redirect and enable the function.

In addition, can be changed the settings for redirection until its function enable operation. The PIOR register can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

**Figure 4-8. Format of Peripheral I/O Redirection Register (PIOR)**

Address: F0077H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PIOR | 0 | 0 | 0 | 0 | 0 | 0 | PIOR1 | PIOR0 |

| Alternate function | 64-pin | | | | 48-pin | | | | 32-pin | | | | 25-pin | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Setting value of PIOR1, PIOR0 | | | | Setting value of PIOR1, PIOR0 | | | | Setting value of PIOR1, PIOR0 | | | | Setting value of PIOR1, PIOR0 | | | |
| | 0, 0 | 0, 1 | 1, 0 | 1, 1 | 0, 0 | 0, 1 | 1, 0 | 1, 1 | 0, 0 | 0, 1 | 1, 0 | 1, 1 | 0, 0 | 0, 1 | 1, 0 | 1, 1 |
| KR0 | P70 | Setting prohibited | P00 | P10 | P70 | Setting prohibited | P02 | P10 | P70 | Setting prohibited | P120 | P10 | – | Setting prohibited | P02 | Setting prohibited |
| KR1 | P71 | | P01 | P11 | P71 | | P03 | P11 | – | | P02 | P11 | – | | P03 | |
| KR2 | P72 | | P02 | P12 | P72 | | P22 | P12 | – | | P03 | P12 | – | | P22 | |
| KR3 | P73 | | P03 | P13 | P73 | | P23 | P13 | – | | P22 | P13 | – | | P23 | |
| KR4 | P74 | | P04 | P14 | P74 | | P24 | P14 | – | | P23 | P14 | – | | – | |
| KR5 | P75 | | P22 | P15 | P75 | | P25 | P15 | – | | P24 | P15 | – | | – | |
| KR6 | P76 | | P23 | P151 | – | | – | – | – | | – | – | – | | – | |
| KR7 | P77 | | P24 | P152 | – | | – | – | – | | – | – | – | | – | |
| KR8 | P05 | | P25 | P153 | – | | – | – | – | | – | – | – | | – | |
| KR9 | P06 | | P26 | P154 | – | | – | – | – | | – | – | – | | – | |

<R> **Remark** –: These functions are not available for use.

### 4.3.9 Global digital input disable register (GDIDIS)

<R> This register is used to prevent through-current flowing to the input buffers of input ports which use $EV_{DD0}$ as the power supply when the $EV_{DD0}$ power supply is turned off.

<R> When not all of the I/O ports using $EV_{DD0}$ as the power supply are used, low power consumption can be achieved by setting the GDIDIS register (to 1) to turn off the $EV_{DD0}$ power supply.

By setting the GDIDIS0 bit to 1, input to any input buffer using $EV_{DD0}$ as the power supply is prohibited, preventing through-current from flowing when the $EV_{DD0}$ power supply is turned off.

The GDIDIS register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

**Remark** GDIDIS register is equipped with 64-pin products.

**Figure 4-9. Format of Global Digital Input Disable Register (GDIDIS)**

Address: F007DH After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| GDIDIS | 0 | 0 | 0 | 0 | 0 | 0 | 0 | GDIDIS0 |

<R>

| GDIDIS0 | Setting of input buffers using $EV_{DD0}$ power supply |
|---|---|
| 0 | Input to input buffers permitted (default) |
| 1 | Input to input buffers prohibited. No through-current flows to the input buffers. |

<R> Turn off the $EV_{DD0}$ power supply with the following procedure.

<1> Prohibit input to input buffers (set GDIDIS0 = 1).

<2> Turn off the $EV_{DD0}$ power supply.

<R> Turn on again the $EV_{DD0}$ power supply with the following procedure.

<1> Turn on the $EV_{DD0}$ power supply.

<2> Permit input to input buffers (set GDIDIS0 = 0).

<R> **Cautions 1. Do not input an input voltage equal to or greater than $EV_{DD0}$ to an input port that uses $EV_{DD}$ as the power supply.**

<R> **2. When input to input buffers is prohibited (GDIDIS0 = 1), the value read from the port register (Pxx) of a port that uses $EV_{DD0}$ as the power supply is 1. When 1 is set in the port output mode register (POMxx) (N-ch open drain output ($EV_{DD0}$ tolerance)), the value read from the port register (Pxx) is 0.**

<R> **Remark** Even when input to input buffers is prohibited (GDIDIS0 = 1), peripheral functions which do not use port functions having $EV_{DD0}$ as the power supply can be used.

### 4.3.10 Global analog input disable register (GAIDIS)

<R> This register is used to prevent through-current flowing from the input buffers of input ports which use $AV_{DD}$ as the power supply when the $AV_{DD}$ power supply is turned off.

<R> When not all of the I/O ports using $AV_{DD}$ as the power supply are used, low power consumption can be achieved by setting the GAIDIS register (to 1) to turn off the $AV_{DD}$ power supply.

By setting the GAIDIS0 bit to 1, input to any input buffer using $AV_{DD}$ as the power supply is prohibited, preventing through-current from flowing when the $AV_{DD}$ power supply is turned off.

When using the GAIDIS register, set the GAIDIS0 bit to 1 while the $AV_{DD}$ voltage is 1.6 V or higher, and then reduce the $AV_{DD}$ voltage to 0 V.

Set GAIDIS0 to 0 after applying power to $AV_{DD}$.

The GAIDIS register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

**Figure 4-10. Format of Global Analog Input Disable Register (GAIDIS)**

Address: F007CH After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| GAIDIS | 0 | 0 | 0 | 0 | 0 | 0 | 0 | GAIDIS0 |

<R>

| GAIDIS0 | Setting of input buffers using $AV_{DD}$ power supply |
|---|---|
| 0 | Input to input buffers permitted (default) |
| 1 | Input to input buffers prohibited. No through-current flows to the input buffers. |

<R> Turn off the $AV_{DD}$ power supply with the following procedure.

<1> Prohibit input to input buffers (set GAIDIS0 = 1).

<2> Turn off the $AV_{DD}$ power supply.

<R> Turn on again the $AV_{DD}$ power supply with the following procedure.

<1> Turn on the $AV_{DD}$ power supply.

<2> Permit input to input buffers (set GAIDIS0 = 0).

<R> **Cautions 1. Do not input an input voltage equal to or greater than $AV_{DD}$ to an input port that uses $AV_{DD}$ as the power supply.**

<R> **2. When input to input buffers is prohibited (GAIDIS0 = 1), the value read from the port register (Pxx) of a port that uses $AV_{DD}$ as the power supply is 1.**

<R> **Remark** Even when input to input buffers is prohibited (GAIDIS0 = 1), peripheral functions which do not use port functions having $EV_{DD}$ as the power supply can be used.

## 4.4 Port Function Operations

Port operations differ depending on whether the input or output mode is set, as shown below.

### 4.4.1 Writing to I/O port

**(1) Output mode**

A value is written to the output latch by a transfer instruction, and the output latch contents are output from the pin.
Once data is written to the output latch, it is retained until data is written to the output latch again.
The data of the output latch is cleared when a reset signal is generated.

**(2) Input mode**

A value is written to the output latch by a transfer instruction, but since the output buffer is off, the pin status does not change. Therefore, byte data can be written to the ports used for both input and output.
Once data is written to the output latch, it is retained until data is written to the output latch again.
The data of the output latch is cleared when a reset signal is generated.

### 4.4.2 Reading from I/O port

**(1) Output mode**

The output latch contents are read by a transfer instruction. The output latch contents do not change.

**(2) Input mode**

The pin status is read by a transfer instruction. The output latch contents do not change.

### 4.4.3 Operations on I/O port

**(1) Output mode**

An operation is performed on the output latch contents, and the result is written to the output latch. The output latch contents are output from the pins.
Once data is written to the output latch, it is retained until data is written to the output latch again.
The data of the output latch is cleared when a reset signal is generated.

**(2) Input mode**

The pin level is read and an operation is performed on its contents. The result of the operation is written to the output latch, but since the output buffer is off, the pin status does not change. Therefore, byte data can be written to the ports used for both input and output.
The data of the output latch is cleared when a reset signal is generated.

### <R> 4.4.4 Handling different potential (1.8 V or 2.5 V) by using $EV_{DD} \leq V_{DD}$

<R> When connecting an external device operating on a different potential (1.8 V or 2.5 V), it is possible to connect the I/O pins of general ports by changing $EV_{DD0}$ to accord with the power supply of the connected device.

### <R> 4.4.5 Handling different potential (1.8 V or 2.5 V) by using I/O buffers

It is possible to connect an external device operating on a different potential (1.8 V or 2.5 V) by switching I/O buffers with the port input mode register (PIMxx) and port output mode register (POMxx).

When receiving input from an external device with a different potential (1.8 V or 2.5 V), set the port input mode registers 0 and 1 (PIM0 and PIM1) on a bit-by-bit basis to enable normal input (CMOS)/TTL input buffer switching.

When outputting data to an external device with a different potential (1.8 V or 2.5 V), set the port output mode registers 0, 1, and 7 (POM0, POM1, and POM7) on a bit-by-bit basis to enable normal output (CMOS)/N-ch open drain ($V_{DD}$ tolerance[Note 1]/$EV_{DD}$ tolerance[Note 2]) switching.

Following, describes the connection of a serial interface.

**Notes 1.** For 25- to 48-pin products

**2.** For 64-pin products

#### <R> (1) Setting procedure when using input ports of UART0 to UART2, CSI00, CSI10, and CSI20 functions for the TTL input buffer

In case of UART0: P11
In case of UART1: P03
In case of UART2: P14
In case of CSI00: P10, P11
In case of CSI10: P03, P04
In case of CSI20: P14, P15

<1> Using an external resistor, pull up externally the input pin to be used to the power supply of the target device (on-chip pull-up resistor cannot be used).

<2> Set the corresponding bit of the PIM0 and PIM1 registers to 1 to switch to the TTL input buffer. For $V_{IH}$ and $V_{IL}$, refer to the DC characteristics when the TTL input buffer is selected.

<3> Enable the operation of the serial array unit and set the mode to the UART/CSI mode.

<R> **(2) Setting procedure when using output ports of UART0 to UART2, CSI00, CSI10, and CSI20 functions in N-ch open-drain output mode**

In case of UART0: P12
In case of UART1: P02
In case of UART2: P13
In case of CSI00: P10, P12
In case of CSI10: P02, P04
In case of CSI20: P13, P15

<1> Using an external resistor, pull up externally the output pin to be used to the power supply of the target device (on-chip pull-up resistor cannot be used).
<2> After reset release, the port mode is the input mode (Hi-Z).
<3> Set the output latch of the corresponding port to 1.
<4> Set the corresponding bit of the POM0 and POM1 registers to 1 to set the N-ch open drain output ($V_{DD}$ tolerance[Note 1]/$EV_{DD}$ tolerance[Note 2]) mode.
<5> Enable the operation of the serial array unit and set the mode to the UART/CSI mode.
<6> Set the output mode by manipulating the PM0 and PM1 registers. At this time, the output data is high level, so the pin is in the Hi-Z state.

**Notes 1.** For 25- to 48-pin products
**2.** For 64-pin products

<R> **(3) Setting procedure when using I/O ports of IIC00, IIC10, and IIC20 functions with a different potential (1.8 V or 2.5 V)**

In case of IIC00: P10, P11
In case of IIC10: P03, P04
In case of IIC20: P14, P15

<1> Using an external resistor, pull up externally the input pin to be used to the power supply of the target device (on-chip pull-up resistor cannot be used).
<2> After reset release, the port mode is the input mode (Hi-Z).
<3> Set the output latch of the corresponding port to 1.
<4> Set the corresponding bit of the POM0 and POM1 registers to 1 to set the N-ch open drain output ($V_{DD}$ tolerance[Note 1]/$EV_{DD}$ tolerance[Note 2]) mode.
<5> Set the corresponding bit of the PIM0 and PIM1 registers to 1 to switch to the TTL input buffer. For $V_{IH}$ and $V_{IL}$, refer to the DC characteristics when the TTL input buffer is selected.
<6> Enable the operation of the serial array unit and set the mode to the simplified $I^2C$ mode.
<7> Set the corresponding bit of the PM0 and PM1 registers to the output mode (data I/O is possible in the output mode). At this time, the output data is high level, so the pin is in the Hi-Z state.

**Notes 1.** For 25- to 48-pin products
**2.** For 64-pin products

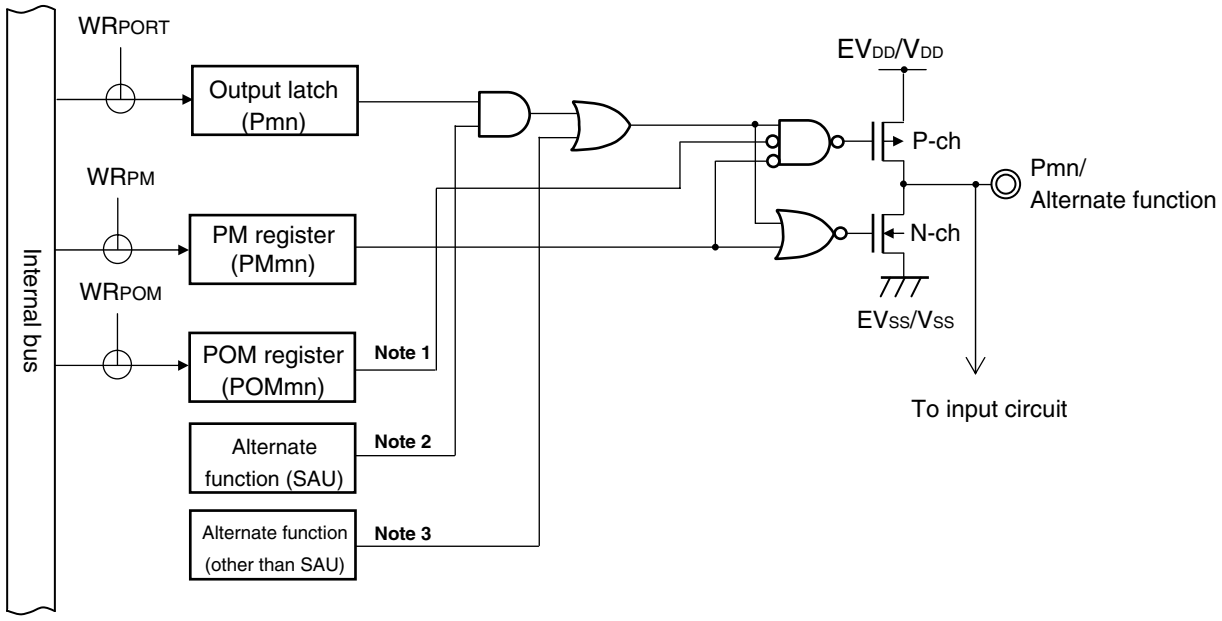
## <R> 4.5 Register Settings When Using Alternate Function

### <R> 4.5.1 Basic concept when using alternate function

In the beginning, for a pin also assigned to be used for analog input, use the ADPC register or port mode control register (PMCxx) to specify whether to use the pin for analog input or digital input/output.

Figure 4-11 shows the basic configuration of an output circuit for pins used for digital input/output. The output of the output latch for the port and the output of the alternate SAU function are input to an AND gate. The output of the AND gate is input to an OR gate. The output of an alternate function other than SAU (TAU, RTC, clock/buzzer output, IICA, etc.) is connected to the other input pin of the OR gate. When such kind of pins are used by the port function or an alternate function, the unused alternate function must not hinder the output of the function to be used. An idea of basic settings for this kind of case is shown in Table 4-5.

**Figure 4-11. Basic Configuration of Output Circuit for Pins**




**Notes 1.** When there is no POM register, this signal should be considered to be low level (0).

**2.** When there is no alternate function, this signal should be considered to be high level (1).

**3.** When there is no alternate function, this signal should be considered to be low level (0).

**Remark** m: Port number (m = 0 to 7, 12 to 15); n: Bit number (n = 0 to 7)

**Table 4-5. Concept of Basic Settings**

| Output Function of Used Pin | Output Settings of Unused Alternate Function | | |
|---|---|---|---|
| | Output Function for Port | Output Function for SAU | Output Function for other than SAU |
| Output function for port | – | Output is high (1) | Output is low (0) |
| Output function for SAU | High (1) | – | Output is low (0) |
| Output function for other than SAU | Low (0) | Output is high (1) | Output is low (0)Note |

**Note** Since more than one output function other than SAU may be assigned to a single pin, the output of an unused alternate function must be set to low level (0). For details on the setting method, see **4.5.2 Register settings for alternate function whose output function is not used**.

<R> **4.5.2 Register settings for alternate function whose output function is not used**

When the output of an alternate function of the pin is not used, the following settings should be made. Note that when the peripheral I/O redirection function is the target, the output can be switched to another pin by setting the peripheral I/O redirection register (PIOR). This allows usage of the port function or other alternate function assigned to the target pin.

(1) SOp = 1, TxDq = 1 (settings when the serial output (SOp/TxDq) of SAU is not used)
When the serial output (SOp/TxDq) is not used, such as, a case in which only the serial input of SAU is used, set the bit in serial output enable register m (SOEm) which corresponds to the unused output to 0 (output disabled) and set the SOmn bit in serial output register m (SOm) to 1 (high). These are the same settings as the initial state.

(2) SCKp = 1, SDAr = 1, SCLr = 1 (settings when channel n in SAU is not used)
When SAU is not used, set bit n (SEmn) in serial channel enable status register m (SEm) to 0 (operation stopped state), set the bit in serial output enable register m (SOEm) which corresponds to the unused output to 0 (output disabled), and set the SOmn and CKOmn bits in serial output register m (SOm) to 1 (high). These are the same settings as the initial state.

(3) TOmn = 0 (settings when the output of channel n in TAU is not used)
When the TOmn output of TAU is not used, set the bit in timer output enable register 0 (TOE0) which corresponds to the unused output to 0 (output disabled) and set the bit in timer output register 0 (TO0) to 0 (low). These are the same settings as the initial state.

(4) SDAAn = 0, SCLAn = 0 (setting when IICA is not used)
When IICA is not used, set the IICEn bit in IICA control register n0 (IICCTLn0) to 0 (operation stopped). This is the same setting as the initial state.

(5) PCLBUZn = 0 (setting when clock/buzzer output is not used)
When the clock/buzzer output is not used, set the PCLOEn bit in clock output select register n (CKSn) to 0 (output disabled). This is the same setting as the initial state.

### <R> 4.5.3 Register setting examples for used port and alternate functions

Register setting examples for used port and alternate functions are shown in Table 4-6. The registers used to control the port functions should be set as shown in Table 4-6. See the following remark for legends used in Table 4-6.

**Remark** –: Not supported
×: don't care
PIORx: Peripheral I/O redirection register
POMxx: Port output mode register
PMCxx: Port mode control register
PMxx: Port mode register
Pxx: Port output latch
Functions in parentheses can be assigned via settings in the peripheral I/O redirection register (PIOR).

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (1/17)**

<R>

| Pin Name | Used Function | | PIOR | POMxx | PMCxx | PMxx | Pxx | Alternate Function Output | | 25-pin | 32-pin | 48-pin | 64-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | SAU Output Function | Other than SAU | | | | |
| P00 | P00 | Input | – | – | – | 1 | × | – | – | × | × | × | √ |
| | | Output | – | – | – | 0 | 0/1 | – | – | | | | |
| | TI00 | Input | × | – | – | 1 | × | – | – | × | × | × | √ |
| | (KR0) | Input | 02H[Note 1] | – | – | 1 | × | – | – | × | × | × | √ |
| P01 | P01 | Input | – | – | – | 1 | × | – | × | × | × | × | √ |
| | | Output | – | – | – | 0 | 0/1 | – | TO00 = 0 | | | | |
| | TO00 | Output | × | – | – | 0 | 0 | – | × | × | × | × | √ |
| | (KR1) | Input | 02H[Note 1] | – | – | 1 | × | – | × | × | × | × | √ |
| P02 | P02 | Input | – | – | 0 | 1 | × | × | – | √ | √ | √ | √ |
| | | Output | – | 0 | 0 | 0 | 0/1 | TxD1 = 1, SO10 = 1[Note 1] | – | | | | |
| | | N-ch open drain output | – | 1 | 0 | 0 | 0/1 | | | | | | |
| | ANI17 | Analog input | × | × | 1 | 1 | × | × | – | √ | √ | √ | √ |
| | TxD1 | Output | × | 0/1 | 0 | 0 | 1 | × | – | √ | √ | √ | √ |
| | TI00 | Input | × | × | 0 | 1 | × | × | – | √ | √ | √ | × |
| | SO10 | Output | × | 0/1 | 0 | 0 | 1 | × | – | × | × | × | √ |
| | (KR0) | Input | 02H[Note 2] | × | 0 | 1 | × | × | – | √ | × | √ | × |
| | (KR1) | Input | 02H[Note 3] | × | 0 | 1 | × | × | – | × | √ | × | × |
| | (KR2) | Input | 02H[Note 1] | × | 0 | 1 | × | × | – | × | × | × | √ |

**Notes 1.** 64-pin products only

**2.** 25- and 48-pin products only

**3.** 32-pin products only

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (2/17)**

<R>

| Pin Name | Used Function | | PIOR | POMxx | PMCxx | PMxx | Pxx | Alternate Function Output | | 25-pin | 32-pin | 48-pin | 64-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | SAU Output Function | Other than SAU | | | | |
| P03 | P03 | Input | – | – | 0 | 1 | × | – | × | √ | √ | √ | √ |
| | | Output | – | 0 | 0 | 0 | 0/1 | – | SDA10 = 1 Note 3<br>TO00 = 0 Note 4 | | | | |
| | | N-ch open drain output | – | 1 | 0 | 0 | 0/1 | – | × | | | | |
| | ANI16 | Analog input | × | × | 1 | 1 | × | – | × | √ | √ | √ | √ |
| | SI10 | Input | × | × | 0 | 1 | × | – | × | × | × | × | √ |
| | RxD1 | Input | × | × | 0 | 1 | × | – | × | √ | √ | √ | √ |
| | SDA10 | I/O | × | 1 | 0 | 0 | 1 | – | × | × | × | × | √ |
| | TO00 | Output | × | 0/1 | 0 | 0 | 1 | – | × | √ | √ | √ | × |
| | (KR1) | Input | 02H Note 4 | × | 0 | 1 | × | – | × | √ | × | √ | × |
| | (KR2) | Input | 02H Note 2 | × | 0 | 1 | × | – | × | × | √ | × | × |
| | (KR3) | Input | 02H Note 3 | × | 0 | 1 | × | – | × | × | × | × | √ |
| P04 | P04 | Input | – | – | – | 1 | × | × | – | × | × | × | √ |
| | | Output | – | 0 | – | 0 | 0/1 | SCK10 = 1, SCL10 = 1 | – | | | | |
| | | N-ch open drain output | – | 1 | – | 0 | 0/1 | | | | | | |
| | SCK10 | Input | × | × | – | 1 | × | × | – | × | × | × | √ |
| | | Output | × | 0/1 | – | 0 | 1 | × | – | × | × | × | √ |
| | SCL10 | Output | × | 0/1 | – | 0 | 1 | × | – | × | × | × | √ |
| | (KR4) | Input | 02H Note 3 | × | – | 1 | × | × | – | × | × | × | √ |
| P05 | P05 | Input | – | – | – | 1 | × | – | × | × | × | × | √ |
| | | Output | – | – | – | 0 | 0/1 | – | TO05 = 0 | | | | |
| | TI05 | Input | × | – | – | 1 | × | – | × | × | × | × | √ |
| | TO05 | Output | × | – | – | 0 | 1 | – | × | × | × | × | √ |
| | KR8 | Input | 00H Note 3 | – | – | 1 | × | – | × | × | × | × | √ |
| P06 | P06 | Input | – | – | – | 1 | × | – | × | × | × | × | √ |
| | | Output | – | – | – | 0 | 0/1 | – | TO06 = 0 | | | | |
| | TI06 | Input | × | – | – | 1 | × | – | × | × | × | × | √ |
| | TO06 | Output | × | – | – | 0 | 1 | – | × | × | × | × | √ |
| | KR9 | Input | 00H Note 3 | – | – | 1 | × | – | × | × | × | × | √ |

**Notes 1.** 25- and 48-pin products only

**2.** 32-pin products only

**3.** 64-pin products only

**4.** 25- to 48-pin products only

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (3/17)**

<R>

| Pin Name | Used Function | | PIOR | POMxx | PMCxx | PMxx | Pxx | Alternate Function Output | | 25-pin | 32-pin | 48-pin | 64-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | SAU Output Function | Other than SAU | | | | |
| P10 | P10 | Input | – | × | – | 1 | × | × | – | √ | √ | √ | √ |
| | | Output | – | 0 | – | 0 | 0/1 | SCK00 = 1, SCL00 = 1 | – | | | | |
| | | N-ch open drain output | – | 1 | – | 0 | 0/1 | × | – | | | | |
| | ANI18 | Analog input | × | × | 1 | 1 | × | × | – | √ | √ | √ | √ |
| | SCK00 | Input | × | × | 0 | 1 | × | × | – | √ | √ | √ | √ |
| | | Output | × | 0/1 | 0 | 0 | 1 | × | – | √ | √ | √ | √ |
| | SCL00 | Output | × | 0/1 | 0 | 0 | 1 | × | – | √ | √ | √ | √ |
| | (KR0) | Input | 03H[Note] | × | 0 | 1 | × | × | – | × | √ | √ | √ |
| P11 | P11 | Input | – | × | – | 1 | × | – | – | √ | √ | √ | √ |
| | | Output | – | 0 | – | 0 | 0/1 | – | SDA00 = 0 | | | | |
| | | N-ch open drain output | – | 1 | – | 0 | 0/1 | | | | | | |
| | ANI20 | Analog input | × | × | 1 | 1 | × | – | × | √ | √ | √ | √ |
| | SI00 | Input | × | × | 0 | 1 | × | – | × | √ | √ | √ | √ |
| | SDA00 | I/O | × | 1 | 0 | 0 | 1 | – | × | √ | √ | √ | √ |
| | RxD0 | Input | × | × | 0 | 1 | × | – | × | √ | √ | √ | √ |
| | TOOLRxD | Input | × | × | 0 | 1 | × | – | × | √ | √ | √ | √ |
| | (KR1) | Input | 03H[Note] | × | 0 | 1 | × | – | × | × | √ | √ | √ |
| P12 | P12 | Input | – | × | – | 1 | × | × | × | √ | √ | √ | √ |
| | | Output | – | 0 | – | 0 | 0/1 | SO00 = 1, TxD0 = 1 | TOOLTxD = 0 | | | | |
| | | N-ch open drain output | – | 1 | – | 0 | 0/1 | × | × | | | | |
| | ANI21 | Analog input | × | × | 1 | 1 | × | × | × | √ | √ | √ | √ |
| | SO00 | Output | × | 0/1 | 0 | 0 | 1 | × | TOOLTxD = 0 | √ | √ | √ | √ |
| | TxD0 | Output | × | 0/1 | 0 | 0 | 1 | × | TOOLTxD = 0 | √ | √ | √ | √ |
| | TOOLTxD | Output | × | 0/1 | 0 | 0 | 1 | SO00 = 1, TxD0 = 1 | × | √ | √ | √ | √ |
| | (KR2) | Input | 03H[Note] | × | 0 | 1 | × | × | × | × | √ | √ | √ |

**Note** 32- to 64-pin products only

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (4/17)**

<R>

| Pin Name | Used Function | | PIOR | POMxx | PMCxx | PMxx | Pxx | Alternate Function Output | | 25-pin | 32-pin | 48-pin | 64-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | SAU Output Function | Other than SAU | | | | |
| P13 | P13 | Input | – | × | – | 1 | × | × | – | × | √ | √ | √ |
| | | Output | – | 0 | – | 0 | 0/1 | SO20 = 1<br>TxD2 = 1 | – | | | | |
| | | N-ch open drain output | – | 1 | – | 0 | 0/1 | × | – | | | | |
| | ANI22 | Analog input | × | × | 1 | 1 | × | × | – | × | √ | √ | √ |
| | SO20 | Output | × | 0/1 | 0 | 0 | 1 | × | – | × | √ | √ | √ |
| | TxD2 | Output | × | 0/1 | 0 | 0 | 1 | × | – | × | √ | √ | √ |
| | (KR3) | Input | 03H[Note] | × | 0 | 1 | × | × | – | × | √ | √ | √ |
| P14 | P14 | Input | – | × | – | 1 | × | – | × | × | √ | √ | √ |
| | | Output | – | 0 | – | 0 | 0/1 | – | SDA20 = 0 | | | | |
| | | N-ch open drain output | – | 1 | – | 0 | 0/1 | | | | | | |
| | ANI23 | Analog input | × | × | 1 | 1 | × | – | × | × | √ | √ | √ |
| | SI20 | Input | × | × | 0 | 1 | × | – | × | × | √ | √ | √ |
| | SDA20 | I/O | × | 1 | 0 | 0 | 1 | – | × | × | √ | √ | √ |
| | RxD2 | Input | × | × | 0 | 1 | × | – | × | × | √ | √ | √ |
| | (KR4) | Input | 03H[Note] | × | 0 | 1 | × | – | × | × | √ | √ | √ |

**Note** 32- to 64-pin products only

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (5/17)**

<R>

| Pin Name | Used Function | | PIOR | POMxx | PMCxx | PMxx | Pxx | Alternate Function Output | | 25-pin | 32-pin | 48-pin | 64-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | SAU Output Function | Other than SAU | | | | |
| P15 | P15 | Input | – | × | – | 1 | × | × | × | × | √ | √ | √ |
| | | Output | – | 0 | – | 0 | 0/1 | SCK20 = 1 Note1 SCL20 = 1 Note 1 | PCLBUZ1 = 0 Note 2 | | | | |
| | | N-ch open drain output | – | 1 | – | 0 | 0/1 | | | | | | |
| | ANI24 | Analog input | × | × | 1 | 1 | × | × | × | × | √ | √ | √ |
| | SCK20 | Input | × | × | 0 | 1 | × | × | × | × | √ | √ | √ |
| | | Output | × | 0/1 | 0 | 0 | 1 | × | PCLBUZ1 = 0 Note 2 | × | √ | √ | √ |
| | SCL20 | Output | × | 0/1 | 0 | 0 | 1 | × | PCLBUZ1 = 0 Note 2 | × | √ | √ | √ |
| | PCLBUZ1 | Output | × | 0 | 0 | 0 | 0 | SCK20 = 1 Note1 SCL20 = 1 Note 1 | × | × | √ | √ | × |
| | (KR5) | Input | 03H Note 1 | × | 0 | 1 | × | × | × | × | √ | √ | √ |
| P16 | P16 | Input | – | × | – | 1 | × | – | × | × | × | √ | √ |
| | | Output | – | 0 | – | 0 | 0/1 | – | TO01 = 0 Note 3 | | | | |
| | TI01 | Input | × | – | – | 1 | × | – | × | × | × | √ | √ |
| | TO01 | Output | × | – | – | 0 | 0 | – | × | × | × | √ | √ |
| | INTP5 | Input | × | – | – | 1 | × | – | × | × | × | √ | √ |

**Notes 1.** 32- to 64-pin products only

**2.** 32- and 48-pin products only

**3.** 48- and 64-pin products only

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (6/17)**

<R>

| Pin Name | Used Function | | ADPC | ADM2 | PIOR | PMxx | Pxx | 25-pin | 32-pin | 48-pin | 64-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | | | | |
| P20 | P20 | Input | ADPC = 01H | × | – | 1 | × | √ | √ | √ | √ |
| | | Output | ADPC = 01H | × | – | 0 | 0/1 | | | | |
| | ANI0 | Analog input | ADPC = 00H/02H to 0FH | 00x0xx0x, 10x0xx0x | × | 1 | × | √ | √ | √ | √ |
| | $AV_{REFP}$ | Reference voltage | ADPC = 00H/02H to 0FH | 01x0xx0x | × | 1 | × | √ | √ | √ | √ |
| P21 | P21 | Input | ADPC = 01H/02H | × | – | 1 | × | √ | √ | √ | √ |
| | | Output | ADPC = 01H/02H | × | – | 0 | 0/1 | | | | |
| | ANI1 | Analog input | ADPC = 00H/03H to 0FH | xx00xx0x | × | 1 | × | √ | √ | √ | √ |
| | $AV_{REFM}$ | Reference voltage | ADPC = 00H/03H to 0FH | xx10xx0x | × | 1 | × | √ | √ | √ | √ |
| P22 | P22 | Input | ADPC = 01H to 03H | × | – | 1 | × | √ | √ | √ | √ |
| | | Output | ADPC = 01H to 03H | × | – | 0 | 0/1 | | | | |
| | ANI2 | Analog input | ADPC = 00H/04H to 0FH | × | × | 1 | × | √ | √ | √ | √ |
| | (KR2) | Input | ADPC = 01H to 03H | × | 02H Note 1 | 1 | × | √ | × | √ | × |
| | (KR3) | Input | ADPC = 01H to 03H | × | 02H Note 2 | 1 | × | × | √ | × | × |
| | (KR5) | Input | ADPC = 01H to 03H | × | 02H Note 3 | 1 | × | × | × | × | √ |
| P23 | P23 | Input | ADPC = 01H to 04H | × | – | 1 | × | √ | √ | √ | √ |
| | | Output | ADPC = 01H to 04H | × | – | 0 | 0/1 | | | | |
| | ANI3 | Analog input | ADPC = 00H/05H to 0FH | × | × | 1 | × | √ | √ | √ | √ |
| | (KR3) | Input | ADPC = 01H to 04H | × | 02H Note 1 | 1 | × | √ | × | √ | × |
| | (KR4) | Input | ADPC = 01H to 04H | × | 02H Note 2 | 1 | × | × | √ | × | × |
| | (KR6) | Input | ADPC = 01H to 04H | × | 02H Note 3 | 1 | × | × | × | × | √ |

**Notes 1.** 25- and 48-pin products only

**2.** 32-pin products only

**3.** 64-pin products only

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (7/17)**

<R>

| Pin Name | Used Function | | ADPC | ADM2 | PIOR | PMxx | Pxx | 25-pin | 32-pin | 48-pin | 64-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | | | | |
| P24 | P24 | Input | ADPC = 01H to 05H | × | – | 1 | × | × | √ | √ | √ |
| | | Output | ADPC = 01H to 05H | × | – | 0 | 0/1 | | | | |
| | ANI4 | Analog input | ADPC = 00H/06H to 0FH | × | × | 1 | × | × | √ | √ | √ |
| | (KR4) | Input | ADPC = 01H to 05H | × | 02H Note 1 | 1 | × | × | × | √ | × |
| | (KR5) | Input | ADPC = 01H to 05H | × | 02H Note 2 | 1 | × | × | √ | × | × |
| | (KR7) | Input | ADPC = 01H to 05H | × | 02H Note 3 | 1 | × | × | × | × | √ |
| P25 | P25 | Input | ADPC = 01H to 06H | × | – | 1 | × | × | × | √ | √ |
| | | Output | ADPC = 01H to 06H | × | – | 0 | 0/1 | | | | |
| | ANI5 | Analog input | ADPC = 00H/07H to 0FH | × | × | 1 | × | × | × | √ | √ |
| | (KR5) | Input | ADPC = 01H to 06H | × | 02H Note 1 | 1 | × | × | × | √ | × |
| | (KR8) | Input | ADPC = 01H to 06H | × | 02H Note 3 | 1 | × | × | × | × | √ |
| P26 | P26 | Input | ADPC = 01H to 07H | × | – | 1 | × | × | × | √ | √ |
| | | Output | ADPC = 01H to 07H | × | – | 0 | 0/1 | | | | |
| | ANI6 | Analog input | ADPC = 00H/08H to 0FH | × | × | 1 | × | × | × | √ | √ |
| | (KR9) | Input | ADPC = 01H to 07H | × | 02H Note 3 | 1 | × | × | × | × | √ |
| P27 | P27 | Input | ADPC = 01H to 08H | × | – | 1 | × | × | × | √ | √ |
| | | Output | ADPC = 01H to 08H | × | – | 0 | 0/1 | | | | |
| | ANI7 | Analog input | ADPC = 00H/09H to 0FH | × | × | 1 | × | × | × | √ | √ |

**Notes 1.** 48-pin products only

**2.** 32-pin products only

**3.** 64-pin products only

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (8/17)**

<R>

| Pin Name | Used Function | | PIOR | POMxx | PMCxx | PMxx | Pxx | Alternate Function Output | | 25-pin | 32-pin | 48-pin | 64-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | SAU Output Function | Other than SAU | | | | |
| P30 | P30 | Input | – | – | – | 1 | × | × | × | √ | √ | √ | √ |
| | | Output | – | – | – | 0 | 0/1 | SCK11 = 1, SCL11 = 1 | RTC1HZ = 0Note 1 | | | | |
| | ANI27 | Analog input | × | – | 1 | 1 | × | × | × | √ | √ | √ | √ |
| | SCK11 | Input | × | – | 0 | 1 | × | × | × | √ | √ | √ | √ |
| | | Output | × | – | 0 | 0 | 1 | × | RTC1HZ = 0Note 1 | √ | √ | √ | √ |
| | SCL11 | Output | × | – | 0 | 0 | 1 | × | RTC1HZ = 0Note 1 | √ | √ | √ | √ |
| | INTP3 | Input | × | – | 0 | 1 | × | × | × | √ | √ | √ | √ |
| | RTC1HZ | Output | × | – | 0 | 0 | 0 | SCK11 = 1, SCL11 = 1 | × | × | × | √ | √ |
| P31 | P31 | Input | – | – | – | 1 | × | – | × | √ | √ | √ | √ |
| | | Output | – | – | – | 0 | 0/1 | – | TO03 = 0, PLCBUZ0 = 0Note 2 | | | | |
| | ANI29 | Analog input | × | – | 1 | 1 | × | – | × | √ | √ | √ | √ |
| | TI03 | Input | × | – | 0 | 1 | × | – | × | √ | √ | √ | √ |
| | TO03 | Output | × | – | 0 | 0 | 0 | – | × | √ | √ | √ | √ |
| | PCLBUZ0 | Output | × | – | 0 | 0 | 0 | – | × | √ | √ | × | × |
| | INTP4 | Input | × | – | 0 | 1 | × | – | × | √ | √ | √ | √ |

**Notes 1.** 48- and 64-pin products only

**2.** 25- and 32-pin products only

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (9/17)**

<R>

| Pin Name | Used Function | | PIOR | POMxx | PMCxx | PMxx | Pxx | Alternate Function Output | | 25-pin | 32-pin | 48-pin | 64-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | SAU Output Function | Other than SAU | | | | |
| P40 | P40 | Input | – | – | – | 1 | × | – | – | √ | √ | √ | √ |
| | | Output | – | – | – | 0 | 0/1 | – | TOOL0 = 1 | | | | |
| | TOOL0 | I/O | × | – | – | × | × | – | × | √ | √ | √ | √ |
| P41 | P41 | Input | – | – | – | 1 | × | – | × | × | × | √ | √ |
| | | Output | – | – | – | 0 | 0/1 | – | TO07 = 0 | | | | |
| | ANI30 | Analog input | × | – | 1 | 1 | × | – | × | × | × | √ | √ |
| | TI07 | Input | × | – | 0 | 1 | × | – | × | × | × | √ | √ |
| | TO07 | Output | × | – | 0 | 0 | 0 | – | × | × | × | √ | √ |
| P42 | P42 | Input | – | – | – | 1 | × | – | × | × | × | × | √ |
| | | Output | – | – | – | 0 | 0/1 | – | TO04 = 0 | | | | |
| | TI04 | Input | × | – | – | 1 | × | – | × | × | × | × | √ |
| | TO04 | Output | × | – | – | 0 | 0 | – | × | × | × | × | √ |
| P43 | P43 | Input | – | – | – | 1 | × | – | – | × | × | × | √ |
| | | Output | – | – | – | 0 | 0/1 | – | – | | | | |

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (10/17)**

<R>

| Pin Name | Used Function | | PIOR | POMxx | PMCxx | PMxx | Pxx | Alternate Function Output | | 25-pin | 32-pin | 48-pin | 64-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | SAU Output Function | Other than SAU | | | | |
| P50 | P50 | Input | – | – | – | 1 | × | × | × | √ | √ | √ | √ |
| | | Output | – | 0 | – | 0 | 0/1 | – | SDA11 = 0 | | | | |
| | | N-ch open drain output | – | 1 | – | 0 | 0/1 | | | | | | |
| | ANI26 | Analog input | × | × | 1 | 1 | × | – | × | √ | √ | √ | √ |
| | SI11 | Input | × | × | 0 | 1 | × | – | × | √ | √ | √ | √ |
| | SDA11 | I/O | × | 1 | 0 | 0 | 1 | – | × | √ | √ | √ | √ |
| | INTP1 | Input | × | × | 0 | 1 | × | – | × | √ | √ | √ | √ |
| P51 | P51 | Input | – | – | – | 1 | × | × | – | √ | √ | √ | √ |
| | | Output | – | – | – | 0 | 0/1 | SO11 = 1 | – | | | | |
| | ANI25 | Analog input | × | – | 1 | 1 | × | × | – | √ | × | √ | √ |
| | SO11 | Output | × | – | 0 | 0 | 1 | × | – | √ | √ | √ | √ |
| | INTP2 | Input | × | – | 0 | 1 | × | × | – | √ | √ | √ | √ |

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (11/17)**

<R>

| Pin Name | Used Function | | PIOR | POMxx | PMCxx | PMxx | Pxx | Alternate Function Output | | 25-pin | 32-pin | 48-pin | 64-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | SAU Output Function | Other than SAU | | | | |
| P60 | P60 | Input | – | – | – | 1 | × | – | × | √ | √ | √ | √ |
| | | N-ch open drain output (6-V tolerance) | – | – | – | 0 | 0/1 | – | SCLA0 = 0 | | | | |
| | SCLA0 | I/O | × | – | – | 0 | 0 | – | × | √ | √ | √ | √ |
| P61 | P61 | Input | – | – | – | 1 | × | – | × | √ | √ | √ | √ |
| | | N-ch open drain output (6-V tolerance) | – | – | – | 0 | 0/1 | – | SDAA0 = 0 | | | | |
| | SDAA0 | I/O | × | – | – | 0 | 0 | – | × | √ | √ | √ | √ |
| P62 | P62 | Input | – | – | – | 1 | × | – | – | × | √ | √ | √ |
| | | N-ch open drain output (6-V tolerance) | – | – | – | 0 | 0/1 | – | – | | | | |
| P63 | P63 | Input | – | – | – | 1 | × | – | – | × | × | √ | √ |
| | | N-ch open drain output (6-V tolerance) | – | – | – | 0 | 0/1 | – | – | | | | |

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (12/17)**

<R>

| Pin Name | Used Function | | PIOR | POMxx | PMCxx | PMxx | Pxx | Alternate Function Output | | 25-pin | 32-pin | 48-pin | 64-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | SAU Output Function | Other than SAU | | | | |
| P70 | P70 | Input | – | – | 0 | 1 | × | × | – | × | √ | √ | √ |
| | | Output | – | – | 0 | 0 | 0/1 | SCK21 = 1[Note 2], SCL21 = 1[Note 2] | – | | | | |
| | ANI28 | Analog input | × | – | 1 | 1 | × | × | – | × | √ | √ | √ |
| | SCK21 | Input | × | – | 0 | 1 | × | × | – | × | × | √ | √ |
| | | Output | × | – | 0 | 0 | 1 | × | – | × | × | √ | √ |
| | SCL21 | Output | × | – | 0 | 0 | 1 | × | – | × | × | √ | √ |
| | KR0 | Input | 00H[Note 1] | – | 0 | 1 | × | × | – | × | √ | √ | √ |
| P71 | P71 | Input | – | – | – | 1 | × | – | – | × | × | √ | √ |
| | | Output | – | 0 | – | 0 | 0/1 | – | SDA21 = 0 | | | | |
| | | N-ch open drain output | – | 1 | – | 0 | 0/1 | | | | | | |
| | SI21 | Input | × | × | – | 1 | × | – | × | × | × | √ | √ |
| | SDA21 | I/O | × | 1 | – | 0 | 1 | – | × | × | × | √ | √ |
| | KR1 | Input | 00H[Note 2] | × | – | 1 | × | – | × | × | × | √ | √ |
| P72 | P72 | Input | – | – | – | 1 | × | × | – | × | × | √ | √ |
| | | Output | – | – | – | 0 | 0/1 | SO21 = 1 | – | | | | |
| | SO21 | Output | × | – | – | 0 | 1 | × | – | × | × | √ | √ |
| | KR2 | Input | 00H[Note 2] | – | – | 1 | × | × | – | × | × | √ | √ |
| P73 | P73 | Input | – | – | – | 1 | × | × | – | × | × | √ | √ |
| | | Output | – | – | – | 0 | 0/1 | SO01 = 1 | – | | | | |
| | SO01 | Output | × | – | – | 0 | 1 | × | – | × | × | √ | √ |
| | KR3 | Input | 00H[Note 2] | – | – | 1 | × | × | – | × | × | √ | √ |

**Notes 1.** 32- to 64-pin products only

**2.** 48- and 64-pin products only

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (13/17)**

<R>

| Pin Name | Used Function | | PIOR | POMxx | PMCxx | PMxx | Pxx | Alternate Function Output | | 25-pin | 32-pin | 48-pin | 64-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | SAU Output Function | Other than SAU | | | | |
| P74 | P74 | Input | – | – | – | 1 | × | – | × | × | × | √ | √ |
| | | Output | – | 0 | – | 0 | 0/1 | – | SDA01 = 0 | | | | |
| | | N-ch open drain output | – | 1 | – | 0 | 0/1 | | | | | | |
| | SI01 | Input | – | × | – | 1 | × | – | × | × | × | √ | √ |
| | SDA01 | I/O | – | 1 | – | 0 | 1 | – | × | × | × | √ | √ |
| | INTP8 | Input | × | × | – | 1 | × | – | × | × | × | √ | √ |
| | KR4 | Input | 00H[Note 1] | × | – | 1 | × | – | × | × | × | √ | √ |
| P75 | P75 | Input | – | – | – | 1 | × | × | – | × | × | √ | √ |
| | | Output | – | – | – | 0 | 0/1 | SCK01 = 1, SCL01 = 1 | – | | | | |
| | SCK01 | Input | × | – | – | 1 | × | × | – | × | × | √ | √ |
| | | Output | × | – | – | 0 | 1 | × | – | × | × | √ | √ |
| | SCL01 | Output | × | – | – | 0 | 1 | × | – | × | × | √ | √ |
| | INTP9 | Input | × | – | – | 1 | × | × | – | × | × | √ | √ |
| | KR5 | Input | 00H[Note 1] | – | – | 1 | × | × | – | × | × | √ | √ |
| P76 | P76 | Input | – | – | – | 1 | × | – | – | × | × | × | √ |
| | | Output | – | – | – | 0 | 0/1 | – | – | | | | |
| | INTP10 | Input | × | – | – | 1 | × | – | – | × | × | × | √ |
| | KR6 | Input | 00H[Note 2] | – | – | 1 | × | – | – | × | × | × | √ |
| P77 | P77 | Input | – | – | – | 1 | × | – | – | × | × | × | √ |
| | | Output | – | – | – | 0 | 0/1 | – | – | | | | |
| | INTP11 | Input | × | – | – | 1 | × | – | – | × | × | × | √ |
| | KR7 | Input | 00H[Note 2] | – | – | 1 | × | – | – | × | × | × | √ |

**Notes 1.** 48- and 64-pin products only

**2.** 64-pin products only

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (14/17)**

<R>

| Pin Name | Used Function | | PIOR | POMxx | PMCxx | PMxx | Pxx | Alternate Function Output | | 25-pin | 32-pin | 48-pin | 64-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | SAU Output Function | Other than SAU | | | | |
| P120 | P120 | Input | – | – | 0 | 1 | × | – | – | × | √ | √ | √ |
| | | Output | – | – | 0 | 0 | 0/1 | – | – | | | | |
| | ANI19 | Analog input | – | – | 1 | 1 | × | – | – | × | √ | √ | √ |
| | (KR0) | Input | 02H[Note] | – | 0 | 1 | × | – | – | × | √ | × | × |

**Note** 32-pin products only

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (15/17)**

<R>

| Pin Name | Used Function | | CMC (EXCLK,OSCSEL, EXCLKS, OSCSELS) | Pxx | 25-pin | 32-pin | 48-pin | 64-pin |
|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | |
| P121 | P121 | Input | 00xx/10 xx/11 xx | × | √ | √ | √ | √ |
| | X1 | – | 01 xx | – | √ | √ | √ | √ |
| P122 | P122 | Input | 00 xx/10 xx | × | √ | √ | √ | √ |
| | X2 | – | 01 xx | – | √ | √ | √ | √ |
| | EXCLK | Input | 11 xx | – | √ | √ | √ | √ |
| P123 | P123 | Input | xx 00/xx 10/xx11 | × | × | × | √ | √ |
| | XT1 | – | xx 01 | – | × | × | √ | √ |
| P124 | P123 | Input | xx 00/xx 10 | × | × | × | √ | √ |
| | XT2 | – | xx 01 | – | × | × | √ | √ |
| | EXCLKS | Input | xx 11 | – | × | × | √ | √ |

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (16/17)**

<R>

| Pin Name | Used Function | | PIOR | POMxx | PMCxx | PMxx | Pxx | Alternate Function Output | | 25-pin | 32-pin | 48-pin | 64-pin |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | SAU Output Function | Other than SAU | | | | |
| P130 | P130 | Output | – | – | – | – | 0/1 | – | – | × | × | √ | √ |
| P137 | P137 | Input | – | – | – | – | × | – | – | √ | √ | √ | √ |
| | INTP0 | Input | × | – | – | – | × | – | – | √ | √ | √ | √ |
| P140 | P140 | Input | – | – | – | 1 | × | – | × | × | × | √ | √ |
| | | Output | – | – | – | 0 | 0/1 | – | PCLBUZ0 = 0 | | | | |
| | PCLBUZ0 | Output | × | – | – | 0 | 0 | – | × | × | × | √ | √ |
| | INTP6 | Input | × | – | – | 1 | × | – | × | × | × | √ | √ |
| P141 | P141 | Input | – | – | – | 1 | × | – | × | × | × | × | √ |
| | | Output | – | – | – | 0 | 0/1 | – | PCLBUZ1 = 0 | | | | |
| | PCLBUZ1 | Output | × | – | – | 0 | 0 | – | × | × | × | × | √ |
| | INTP7 | Input | × | – | – | 1 | × | – | × | × | × | × | √ |

**Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function (17/17)**

<R>

| Pin Name | Used Function | | ADPC | PIOR | PMxx | Pxx | 25-pin | 32-pin | 48-pin | 64-pin |
|---|---|---|---|---|---|---|---|---|---|---|
| | Function Name | I/O | | | | | | | | |
| P150 | P150 | Input | ADPC = 01H to 09H | – | 1 | × | × | × | √ | √ |
| | | Output | ADPC = 01H to 09H | – | 0 | 0/1 | | | | |
| | ANI8 | Analog input | ADPC = 00H/0AH to 0FH | × | 1 | × | × | × | √ | √ |
| P151 | P151 | Input | ADPC = 01H to 0AH | – | 1 | × | × | × | × | √ |
| | | Output | ADPC = 01H to 0AH | – | 0 | 0/1 | | | | |
| | ANI9 | Analog input | ADPC = 00H/0BH to 0FH | × | 1 | × | × | × | × | √ |
| | (KR6) | Input | ADPC = 01H to 0AH | 03H[Note] | 1 | × | × | × | × | √ |
| P152 | P152 | Input | ADPC = 01H to 0BH | – | 1 | × | × | × | × | √ |
| | | Output | ADPC = 01H to 0BH | – | 0 | 0/1 | | | | |
| | ANI10 | Analog input | ADPC = 00H/0CH to 0FH | × | 1 | × | × | × | × | √ |
| | (KR7) | Input | ADPC = 01H to 0BH | 03H[Note] | 1 | × | × | × | × | √ |
| P153 | P153 | Input | ADPC = 01H to 0CH | – | 1 | × | × | × | × | √ |
| | | Output | ADPC = 01H to 0CH | – | 0 | 0/1 | | | | |
| | ANI11 | Analog input | ADPC = 00H/0DH to 0FH | × | 1 | × | × | × | × | √ |
| | (KR8) | Input | ADPC = 01H to 0CH | 03H[Note] | 1 | × | × | × | × | √ |
| P154 | P154 | Input | ADPC = 01H to 0DH | – | 1 | × | × | × | × | √ |
| | | Output | ADPC = 01H to 0DH | – | 0 | 0/1 | | | | |
| | ANI12 | Analog input | ADPC = 00H/0EH/0FH | × | 1 | × | × | × | × | √ |
| | (KR9) | Input | ADPC = 01H to 0DH | 03H[Note] | 1 | × | × | × | × | √ |

**Note** 64-pin products only

## 4.6 Cautions When Using Port Function

### 4.6.1 Cautions on 1-bit manipulation instruction for port register n (Pn)

When a 1-bit manipulation instruction is executed on a port that provides both input and output functions, the output latch value of an input port that is not subject to manipulation may be written in addition to the targeted bit.

Therefore, it is recommended to rewrite the output latch when switching a port from input mode to output mode.

<Example> When P10 is an output port, P11 to P16 are input ports (all pin statuses are high level), and the port latch value of port 1 is 00H, if the output of output port P10 is changed from low level to high level via a 1-bit manipulation instruction, the output latch value of port 1 is FFH.

Explanation: The targets of writing to and reading from the Pn register of a port whose PMnm bit is 1 are the output latch and pin status, respectively.

A 1-bit manipulation instruction is executed in the following order in the RL78/G1A.

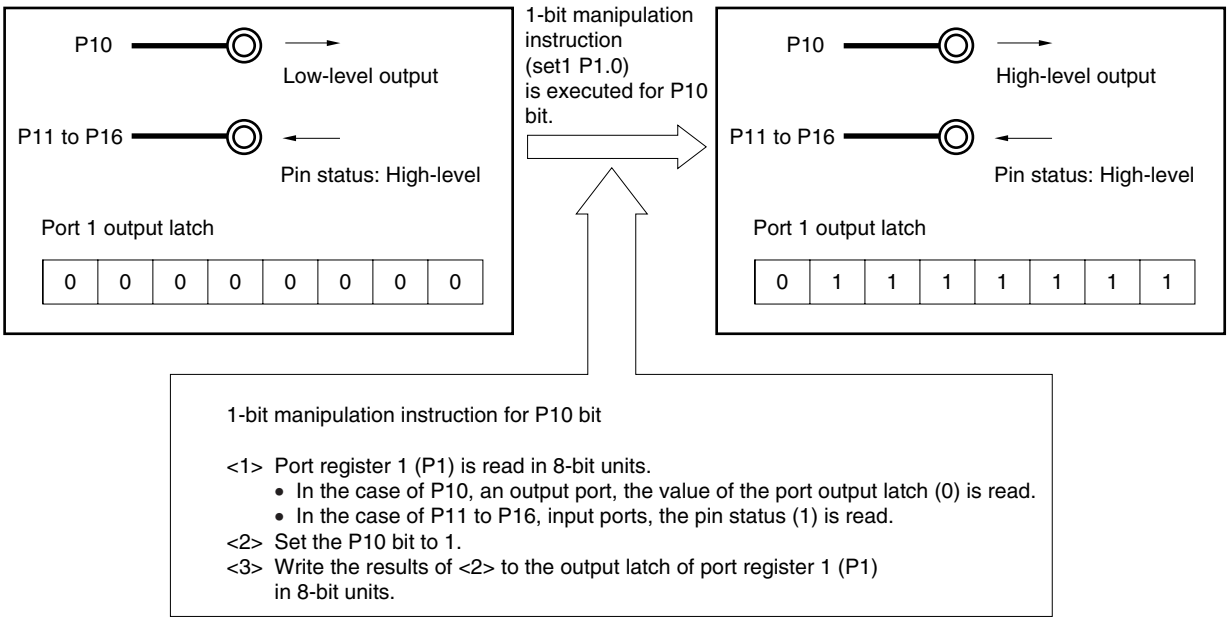
<1> The Pn register is read in 8-bit units.
<2> The targeted one bit is manipulated.
<3> The Pn register is written in 8-bit units.

In step <1>, the output latch value (0) of P10, which is an output port, is read, while the pin statuses of P11 to P16, which are input ports, are read. If the pin statuses of P11 to P16 are high level at this time, the read value is FEH.

The value is changed to FFH by the manipulation in <2>.

FFH is written to the output latch by the manipulation in <3>.

**Figure 4-12. Bit Manipulation Instruction (P10)**



### 4.6.2 Notes on specifying the pin settings

<R> For an output pin to which multiple functions are assigned, the output of the unused alternate functions must be set to its initial state so as to prevent conflicting outputs. This also applies to the functions assigned by using the peripheral I/O redirection register (PIOR). For details about the alternate function output, see **4.5 Register Settings When Using Alternate Function**.

No specific setting is required for input pins because the output of their alternate functions is disabled (the buffer output is Hi-Z).

Disabling the unused functions, including blocks that are only used for input or do not have I/O, is recommended for lower power consumption.

# CHAPTER 5 CLOCK GENERATOR

The presence or absence of connecting resonator pin for main system clock, connecting resonator pin for subsystem clock, external clock input pin for main system clock, and external clock input pin for subsystem clock, depends on the product.

| Output pin | 25, 32-pin | 48, 64-pin |
|---|---|---|
| X1, X2 pins | √ | √ |
| EXCLK pin | √ | √ |
| XT1, XT2 pins | – | √ |
| EXCLKS pin | – | √ |

## 5.1 Functions of Clock Generator

The clock generator generates the clock to be supplied to the CPU and peripheral hardware.
The following three kinds of system clocks and clock oscillators are selectable.

**(1) Main system clock**

**<1> X1 oscillator**

This circuit oscillates a clock of $f_X$ = 1 to 20 MHz by connecting a resonator to X1 and X2.
Oscillation can be stopped by executing the STOP instruction or setting of the MSTOP bit (bit 7 of the clock operation status control register (CSC)).

**<2> High-speed on-chip oscillator**

The frequency at which to oscillate can be selected from among $f_{IH}$ = 32, 24, 16, 12, 8, 6, 4, 3, 2, or 1 MHz (typ.) by using the option byte (000C2H). After a reset release, the CPU always starts operating with this high-speed on-chip oscillator clock. Oscillation can be stopped by executing the STOP instruction or setting the HIOSTOP bit (bit 0 of the CSC register).
The frequency specified by using an option byte can be changed by using the high-speed on-chip oscillator frequency select register (HOCODIV). For details about the frequency, see Figure 5-9 Format of High-speed On-chip Oscillator Frequency Select Register (HOCODIV).
The frequencies that can be specified for the high-speed on-chip oscillator by using the option byte and the high-speed on-chip oscillator frequency select register (HOCODIV) are shown below.

| Power Supply Voltage | Flash Operation Mode | Oscillation Frequency (MHz) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 6 | 8 | 12 | 16 | 24 | 32 |
| 2.7 V ≤ $V_{DD}$ ≤ 3.6 V | HS (high-speed main) mode | √ | √ | √ | √ | √ | √ | √ | √ | √ | √ |
| 2.4 V ≤ $V_{DD}$ ≤ 3.6 V | | √ | √ | √ | √ | √ | √ | √ | √ | – | – |
| 1.8 V ≤ $V_{DD}$ ≤ 3.6 V | LS (low-speed main) mode | √ | √ | √ | √ | √ | √ | – | – | – | – |
| 1.6 V ≤ $V_{DD}$ ≤ 3.6 V | LV (low-voltage main) mode | √ | √ | √ | √ | – | – | – | – | – | – |

An external main system clock ($f_{EX}$ = 1 to 20 MHz) can also be supplied from the EXCLK/X2/P122 pin. An external main system clock input can be disabled by executing the STOP instruction or setting of the MSTOP bit.
As the main system clock, a high-speed system clock (X1 clock or external main system clock) or high-speed on-chip oscillator clock can be selected by setting of the MCM0 bit (bit 4 of the system clock control register (CKC)).

**(2) Subsystem clock**

- **XT1 clock oscillator**

  This circuit oscillates a clock of $f_{XT}$ = 32.768 kHz by connecting a 32.768 kHz resonator to XT1 and XT2. Oscillation can be stopped by setting the XTSTOP bit (bit 6 of the clock operation status control register (CSC)).

An external subsystem clock ($f_{EXS}$ = 32.768 KHz) can also be supplied from the EXCLKS/XT2/P124 pin. An external subsystem clock input can be disabled by the setting of the XTSTOP bit.

**(3) Low-speed on-chip oscillator clock (Low-speed on-chip oscillator)**

This circuit oscillates a clock of $f_{IL}$ = 15 kHz (TYP.).
The low-speed on-chip oscillator clock cannot be used as the CPU clock.
Only the following peripheral hardware runs on the low-speed on-chip oscillator clock.

- Watchdog timer
- Real-time clock
- 12-bit interval timer

<R> This clock operates when bit 4 (WDTON) of the option byte (000C0H), bit 4 (WUTMMCK0) of the subsystem clock supply mode control register (OSMC), or both are set to 1.
However, when WDTON = 1, WUTMMCK0 = 0, and bit 0 (WDSTBYON) of the option byte (000C0H) is 0, oscillation of the low-speed on-chip oscillator stops if the HALT or STOP instruction is executed.

<R> **Caution The low-speed on-chip oscillator clock ($f_{IL}$) can only be selected as the real-time clock count clock when the fixed-cycle interrupt function is used.**

**Remark** $f_X$: X1 clock oscillation frequency
$f_{IH}$: High-speed on-chip oscillator clock frequency
$f_{EX}$: External main system clock frequency
$f_{XT}$: XT1 clock oscillation frequency
$f_{EXS}$: External subsystem clock frequency
$f_{IL}$: Low-speed on-chip oscillator clock frequency

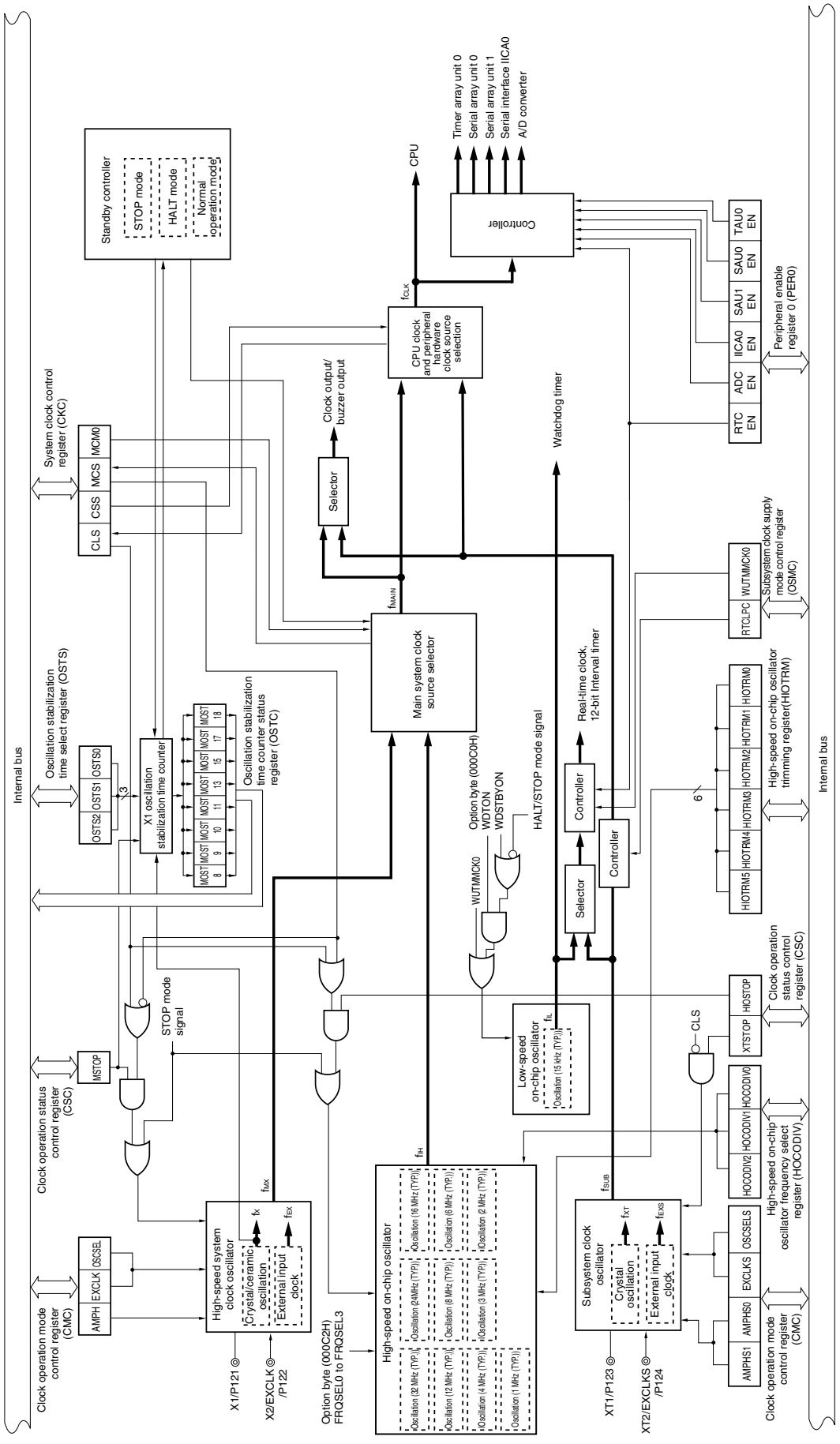
## 5.2 Configuration of Clock Generator

The clock generator includes the following hardware.

**Table 5-1. Configuration of Clock Generator**

| Item | Configuration |
|---|---|
| Control registers | Clock operation mode control register (CMC)<br>System clock control register (CKC)<br>Clock operation status control register (CSC)<br>Oscillation stabilization time counter status register (OSTC)<br>Oscillation stabilization time select register (OSTS)<br>Peripheral enable register 0 (PER0)<br><R> Subsystem clock supply mode control register (OSMC)<br>High-speed on-chip oscillator frequency select register (HOCODIV)<br>High-speed on-chip oscillator trimming register (HIOTRM) |
| Oscillators | X1 oscillator<br>XT1 oscillator<br>High-speed on-chip oscillator<br>Low-speed on-chip oscillator |

**Figure 5-1. Block Diagram of Clock Generator**




<R>

(**Remark** is listed on the next page after next.)

**Remark** $f_X$: X1 clock oscillation frequency
$f_{IH}$: High-speed on-chip oscillator clock frequency
$f_{EX}$: External main system clock frequency
$f_{MX}$: High-speed system clock frequency
$f_{MAIN}$: Main system clock frequency
$f_{XT}$: XT1 clock oscillation frequency
$f_{EXS}$: External subsystem clock frequency
$f_{SUB}$: Subsystem clock frequency
$f_{CLK}$: CPU/peripheral hardware clock frequency
$f_{IL}$: Low-speed on-chip oscillator clock frequency

## 5.3 Registers Controlling Clock Generator

The following nine registers are used to control the clock generator.

- Clock operation mode control register (CMC)
- System clock control register (CKC)
- Clock operation status control register (CSC)
- Oscillation stabilization time counter status register (OSTC)
- Oscillation stabilization time select register (OSTS)
- Peripheral enable register 0 (PER0)
<R> - Subsystem clock supply mode control register (OSMC)
- High-speed on-chip oscillator frequency select register (HOCODIV)
- High-speed on-chip oscillator trimming register (HIOTRM)

<R> **Caution Which registers and bits are included depends on the product. Be sure to set registers and bits that are not mounted in a product to their initial values.**

### 5.3.1 Clock operation mode control register (CMC)

This register is used to set the operation mode of the X1/P121, X2/EXCLK/P122, XT1/P123, and XT2/EXCLKS/P124 pins, and to select a gain of the oscillator.

The CMC register can be written only once by an 8-bit memory manipulation instruction after reset release. This register can be read by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 5-2. Format of Clock Operation Mode Control Register (CMC)**

Address: FFFA0H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CMC | EXCLK | OSCSEL | EXCLKS | OSCSELS | 0 | AMPHS1 | AMPHS0 | AMPH |

| EXCLK | OSCSEL | High-speed system clock pin operation mode | X1/P121 pin | X2/EXCLK/P122 pin |
|---|---|---|---|---|
| 0 | 0 | Input port mode | Input port | |
| 0 | 1 | X1 oscillation mode | Crystal/ceramic resonator connection | |
| 1 | 0 | Input port mode | Input port | |
| 1 | 1 | External clock input mode | Input port | External clock input |

| EXCLKS | OSCSELS | Subsystem clock pin operation mode | XT1/P123 pin | XT2/EXCLKS/P124 pin |
|---|---|---|---|---|
| 0 | 0 | Input port mode | Input port | |
| 0 | 1 | XT1 oscillation mode | Crystal resonator connection | |
| 1 | 0 | Input port mode | Input port | |
| 1 | 1 | External clock input mode | Input port | External clock input |

| AMPHS1 | AMPHS0 | XT1 oscillator oscillation mode selection |
|---|---|---|
| 0 | 0 | Low power consumption oscillation (default) |
| 0 | 1 | Normal oscillation |
| 1 | 0 | Ultra-low power consumption oscillation |
| 1 | 1 | Setting prohibited |

| AMPH | Control of X1 clock oscillation frequency |
|---|---|
| 0 | 1 MHz ≤ $f_X$ ≤ 10 MHz |
| 1 | 10 MHz < $f_X$ ≤ 20 MHz |

**Cautions 1. The CMC register can be written only once after reset release, by an 8-bit memory manipulation instruction. When using the CMC register with its initial value (00H), be sure to set the register to 00H after a reset ends in order to prevent malfunction due to a program loop. Such a malfunction becomes unrecoverable when a value other than 00H is mistakenly written.**

**2. After reset release, set the CMC register before X1 or XT1 oscillation is started as set by the clock operation status control register (CSC).**

**3. Be sure to set the AMPH bit to 1 if the X1 clock oscillation frequency exceeds 10 MHz.**

**4. Specify the settings for the AMPH, AMPHS1, and AMPHS0 bits while $f_{IH}$ is selected as $f_{CLK}$ after a reset ends (before $f_{CLK}$ is switched to $f_{MX}$ or $f_{SUB}$).**

**5. Oscillation stabilization time of $f_{XT}$, counting on the software.**

**6. Although the maximum system clock frequency is 32 MHz, the maximum frequency of the X1 oscillator is 20 MHz.**

(**Cautions** and **Remark** are given on the next page.)

**Cautions 7. The XT1 oscillator is a circuit with low amplification in order to achieve low-power consumption. Note the following points when designing the circuit.**

- **Pins and circuit boards include parasitic capacitance. Therefore, perform oscillation evaluation using a circuit board to be actually used and confirm that there are no problems.**
- <R> **Before using the ultra-low power consumption oscillation (AMPHS1, AMPHS0 = 1, 0) as the mode of the XT1 oscillator, evaluate the resonators described in 5.7 Resonator and Oscillator Constants.**
- **Make the wiring between the XT1 and XT2 pins and the resonators as short as possible, and minimize the parasitic capacitance and wiring resistance. Note this particularly when the ultra-low power consumption oscillation (AMPHS1, AMPHS0 = 1, 0) is selected.**
- **Configure the circuit of the circuit board, using material with little wiring resistance.**
- **Place a ground pattern that has the same potential as $V_{SS}$ as much as possible near the XT1 oscillator.**
- **Be sure that the signal lines between the XT1 and XT2 pins, and the resonators do not cross with the other signal lines. Do not route the wiring near a signal line through which a high fluctuating current flows.**
- **The impedance between the XT1 and XT2 pins may drop and oscillation may be disturbed due to moisture absorption of the circuit board in a high-humidity environment or dew condensation on the board. When using the circuit board in such an environment, take measures to damp-proof the circuit board, such as by coating.**
- **When coating the circuit board, use material that does not cause capacitance or leakage between the XT1 and XT2 pins.**

**Remark** fx: X1 clock frequency

### 5.3.2 System clock control register (CKC)

This register is used to select a CPU/peripheral hardware clock and a main system clock.

The CKC register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

**Figure 5-3. Format of System Clock Control Register (CKC)**

Address: FFFA4H After reset: 00H R/W[Note 1]

| Symbol | <7> | <6> | <5> | <4> | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CKC | CLS | CSS | MCS | MCM0 | 0 | 0 | 0 | 0 |

| CLS | Status of CPU/peripheral hardware clock ($f_{CLK}$) |
|---|---|
| 0 | Main system clock ($f_{MAIN}$) |
| 1 | Subsystem clock ($f_{SUB}$) |

| CSS | Selection of CPU/peripheral hardware clock ($f_{CLK}$) |
|---|---|
| 0 | Main system clock ($f_{MAIN}$) |
| 1 [Note 2] | Subsystem clock ($f_{SUB}$) |

| MCS | Status of Main system clock ($f_{MAIN}$) |
|---|---|
| 0 | High-speed on-chip oscillator clock ($f_{IH}$) |
| 1 | High-speed system clock ($f_{MX}$) |

| MCM0 [Note 2] | Main system clock ($f_{MAIN}$) operation control |
|---|---|
| 0 | Selects the high-speed on-chip oscillator clock ($f_{IH}$) as the main system clock ($f_{MAIN}$) |
| 1 | Selects the high-speed system clock ($f_{MX}$) as the main system clock ($f_{MAIN}$) |

**Notes 1.** Bits 7 and 5 are read-only.

**2.** Changing the value of the MCM0 bit is prohibited while the CSS bit is set to 1.

**Cautions 1. Be sure to set bit 3 to 0 to 0.**

**2. The clock set by the CSS bit is supplied to the CPU and peripheral hardware. If the CPU clock is changed, therefore, the clock supplied to peripheral hardware (except the real-time clock, 12-bit interval timer, clock output/buzzer output, and watchdog timer) is also changed at the same time. Consequently, stop each peripheral function when changing the CPU/peripheral hardware clock.**

**3. If the subsystem clock is used as the peripheral hardware clock, the operations of the A/D converter and IICA are not guaranteed. For the operating characteristics of the peripheral hardware, refer to the chapters describing the various peripheral hardware as well as CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C), CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to + 105°C).**

**Remark** $f_{IH}$: High-speed on-chip oscillator clock frequency

$f_{MX}$: High-speed system clock frequency

$f_{MAIN}$: Main system clock frequency

$f_{SUB}$: Subsystem clock frequency

### 5.3.3 Clock operation status control register (CSC)

This register is used to control the operations of the high-speed system clock, high-speed on-chip oscillator clock, and subsystem clock (except the low-speed on-chip oscillator clock).

The CSC register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to C0H.

**Figure 5-4. Format of Clock Operation Status Control Register (CSC)**

Address: FFFA1H After reset: C0H R/W

| Symbol | <7> | <6> | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| CSC | MSTOP | XTSTOP | 0 | 0 | 0 | 0 | 0 | HIOSTOP |

| MSTOP | High-speed system clock operation control | | |
|---|---|---|---|
| | X1 oscillation mode | External clock input mode | Input port mode |
| 0 | X1 oscillator operating | External clock from EXCLK pin is valid | Input port |
| 1 | X1 oscillator stopped | External clock from EXCLK pin is invalid | |

| XTSTOP | Subsystem clock operation control | | |
|---|---|---|---|
| | XT1 oscillation mode | External clock input mode | Input port mode |
| 0 | XT1 oscillator operating | External clock from EXCLKS pin is valid | Input port |
| 1 | XT1 oscillator stopped | External clock from EXCLKS pin is invalid | |

| HIOSTOP | High-speed on-chip oscillator clock operation control |
|---|---|
| 0 | High-speed on-chip oscillator operating |
| 1 | High-speed on-chip oscillator stopped |

**Cautions 1. After reset release, set the clock operation mode control register (CMC) before setting the CSC register.**

**2. Set the oscillation stabilization time select register (OSTS) before setting the MSTOP bit to 0 after releasing reset. Note that if the OSTS register is being used with its default settings, the OSTS register is not required to be set here.**

**3. To start X1 oscillation as set by the MSTOP bit, check the oscillation stabilization time of the X1 clock by using the oscillation stabilization time counter status register (OSTC).**

**4. When starting XT1 oscillation by setting the XTSTOP bit, wait for oscillation of the subsystem clock to stabilize by setting a wait time using software.**

**5. Do not stop the clock selected for the CPU peripheral hardware clock ($f_{CLK}$) with the OSC register.**

**6. The setting of the flags of the register to stop clock oscillation (invalidate the external clock input) and the condition before clock oscillation is to be stopped are as Table 5-2.**

**Table 5-2. Stopping Clock Method**

| Clock | Condition Before Stopping Clock (Invalidating External Clock Input) | Setting of CSC Register Flags |
|---|---|---|
| X1 clock<br>External main system clock | CPU and peripheral hardware clocks operate with a clock other than the high-speed system clock.<br>(CLS = 0 and MCS = 0, or CLS = 1) | MSTOP = 1 |
| XT1 clock<br>External subsystem clock | CPU and peripheral hardware clocks operate with a clock other than the subsystem clock.<br>(CLS = 0) | XTSTOP = 1 |
| High-speed on-chip oscillator clock | CPU and peripheral hardware clocks operate with a clock other than the high-speed on-chip oscillator clock.<br>(CLS = 0 and MCS = 1, or CLS = 1) | HIOSTOP = 1 |

### 5.3.4 Oscillation stabilization time counter status register (OSTC)

This is the register that indicates the count status of the X1 clock oscillation stabilization time counter.

The X1 clock oscillation stabilization time can be checked in the following case,

- If the X1 clock starts oscillation while the high-speed on-chip oscillator clock or subsystem clock is being used as the CPU clock.
- If the STOP mode is entered and then released while the high-speed on-chip oscillator clock is being used as the CPU clock with the X1 clock oscillating.

The OSTC register can be read by a 1-bit or 8-bit memory manipulation instruction.

When reset signal is generated, the STOP instruction and MSTOP (bit 7 of clock operation status control register (CSC)) = 1 clear the OSTC register to 00H.

**Remark** The oscillation stabilization time counter starts counting in the following cases.

- When oscillation of the X1 clock starts (EXCLK, OSCSEL = 0, 1 → MSTOP = 0)
- When the STOP mode is released

**Figure 5-5. Format of Oscillation Stabilization Time Counter Status Register (OSTC)**

Address: FFFA2H After reset: 00H R

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| OSTC | MOST 8 | MOST 9 | MOST 10 | MOST 11 | MOST 13 | MOST 15 | MOST 17 | MOST 18 |

<R>

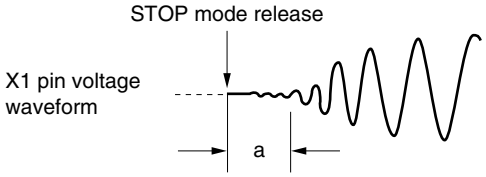
| MOST 8 | MOST 9 | MOST 10 | MOST 11 | MOST 13 | MOST 15 | MOST 17 | MOST 18 | Oscillation stabilization time status | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | fx = 10 MHz | fx = 20 MHz |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | $2^8$/fx max. | 25.6 $\mu$s max. | 12.8 $\mu$s max. |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | $2^8$/fx min. | 25.6 $\mu$s min. | 12.8 $\mu$s min. |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | $2^9$/fx min. | 51.2 $\mu$s min. | 25.6 $\mu$s min. |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | $2^{10}$/fx min. | 102 $\mu$s min. | 51.2 $\mu$s min. |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | $2^{11}$/fx min. | 204 $\mu$s min. | 102 $\mu$s min. |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | $2^{13}$/fx min. | 819 $\mu$s min. | 409 $\mu$s min. |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | $2^{15}$/fx min. | 3.27 ms min. | 1.63 ms min. |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | $2^{17}$/fx min. | 13.1 ms min. | 6.55 ms min. |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | $2^{18}$/fx min. | 26.2 ms min. | 13.1 ms min. |

**Cautions 1. After the above time has elapsed, the bits are set to 1 in order from the MOST8 bit and remain 1.**

**2. The oscillation stabilization time counter counts up to the oscillation stabilization time set by the oscillation stabilization time select register (OSTS).**
**In the following cases, set the oscillation stabilization time of the OSTS register to the value greater than the count value which is to be checked by the OSTC register.**

- **If the X1 clock starts oscillation while the high-speed on-chip oscillator clock or subsystem clock is being used as the CPU clock.**
- **If the STOP mode is entered and then released while the high-speed on-chip oscillator clock is being used as the CPU clock with the X1 clock oscillating.**
**(Note, therefore, that only the status up to the oscillation stabilization time set by the OSTS register is set to the OSTC register after the STOP mode is released.)**

**3. The X1 clock oscillation stabilization wait time does not include the time until clock oscillation starts (“a” below).**




**Remark** fx: X1 clock oscillation frequency

### 5.3.5 Oscillation stabilization time select register (OSTS)

<R> This register is used to select the X1 clock oscillation stabilization wait time.

<R> When the X1 clock is made to oscillate by clearing the MSTOP bit to start the X1 oscillation circuit operating, actual operation is automatically delayed for the time set in the OSTS register.

<R> When switching the CPU clock from the high-speed on-chip oscillator clock or the subsystem clock to the X1 clock, and when using the high-speed on-chip oscillator clock for switching the X1 clock from the oscillating state to STOP mode, use the oscillation stabilization time counter status register (OSTC) to confirm that the desired oscillation stabilization time has elapsed after release from the STOP mode. That is, use the OSTC register to check that the oscillation stabilization time corresponding to its setting has been reached

The OSTS register can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets the OSTS register to 07H.

**Figure 5-6. Format of Oscillation Stabilization Time Select Register (OSTS)**

Address: FFFA3H After reset: 07H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| OSTS | 0 | 0 | 0 | 0 | 0 | OSTS2 | OSTS1 | OSTS0 |

<R>

| OSTS2 | OSTS1 | OSTS0 | Oscillation stabilization time selection | | |
|---|---|---|---|---|---|
| | | | | $f_X$ = 10 MHz | $f_X$ = 20 MHz |
| 0 | 0 | 0 | $2^8/f_X$ | 25.6 $\mu$s | 12.8 $\mu$s |
| 0 | 0 | 1 | $2^9/f_X$ | 51.2 $\mu$s | 25.6 $\mu$s |
| 0 | 1 | 0 | $2^{10}/f_X$ | 102 $\mu$s | 51.2 $\mu$s |
| 0 | 1 | 1 | $2^{11}/f_X$ | 204 $\mu$s | 102 $\mu$s |
| 1 | 0 | 0 | $2^{13}/f_X$ | 819 $\mu$s | 409 $\mu$s |
| 1 | 0 | 1 | $2^{15}/f_X$ | 3.27 ms | 1.63 ms |
| 1 | 1 | 0 | $2^{17}/f_X$ | 13.1 ms | 6.55 ms |
| 1 | 1 | 1 | $2^{18}/f_X$ | 26.2 ms | 13.1 ms |

<R>

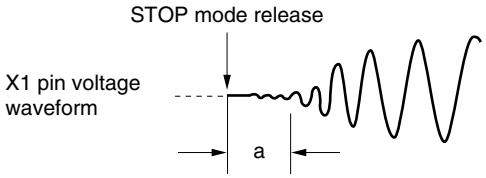
**Cautions 1. Change the setting of the OSTS register before setting the MSTOP bit of the clock operation status control register (CSC) to 0.**

**2. The oscillation stabilization time counter counts up to the oscillation stabilization time set by the OSTS register.**

**In the following cases, set the oscillation stabilization time of the OSTS register to the value greater than the count value which is to be checked by the OSTC register after the oscillation starts.**

- **If the X1 clock starts oscillation while the high-speed on-chip oscillator clock or subsystem clock is being used as the CPU clock.**
- **If the STOP mode is entered and then released while the high-speed on-chip oscillator clock is being used as the CPU clock with the X1 clock oscillating. (Note, therefore, that only the status up to the oscillation stabilization time set by the OSTS register is set to the OSTC register after the STOP mode is released.)**

**3. The X1 clock oscillation stabilization wait time does not include the time until clock oscillation starts ("a" below).**




**Remark** fx: X1 clock oscillation frequency

### 5.3.6 Peripheral enable register 0 (PER0)

These registers are used to enable or disable supplying the clock to the peripheral hardware. Clock supply to the hardware that is not used is also stopped so as to decrease the power consumption and noise.

To use the peripheral functions below, which are controlled by this register, set (1) the bit corresponding to each function before specifying the initial settings of the peripheral functions.

- Real-time clock, 12-bit interval timer
- A/D converter
- Serial interface IICA0
- Serial array unit 1
- Serial array unit 0
- Timer array unit 0

The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H.

**Figure 5-7. Format of Peripheral Enable Register 0 (PER0) (1/3)**

Address: F00F0H After reset: 00H R/W

| Symbol | <7> | 6 | <5> | <4> | <3> | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER0 | RTCEN | 0 | ADCEN | IICA0EN | SAU1EN | SAU0EN | 0 | TAU0EN |

| RTCEN | Control of real-time clock (RTC) and 12-bit interval timer input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFR used by the real-time clock (RTC) and 12-bit interval timer cannot be written.<br>• The real-time clock (RTC) and 12-bit interval timer are in the reset status. |
| 1 | Enables input clock supply.<br>• SFR used by the real-time clock (RTC) and 12-bit interval timer can be read and written. |

**Caution Be sure to clear the following bits to 0.**
**25-pin products: bits 1, 3, 6**
**32, 48, 64-pin products: bits 1, 6**

**Figure 5-7. Format of Peripheral Enable Register 0 (PER0) (2/3)**

Address: F00F0H After reset: 00H R/W

| Symbol | <7> | 6 | <5> | <4> | <3> | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER0 | RTCEN | 0 | ADCEN | IICA0EN | SAU1EN | SAU0EN | 0 | TAU0EN |

| ADCEN | Control of A/D converter input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFR used by the A/D converter cannot be written.<br>• The A/D converter is in the reset status. |
| 1 | Enables input clock supply.<br>• SFR used by the A/D converter can be read and written. |

| IICA0EN | Control of serial interface IICA0 input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFR used by the serial interface IICA0 cannot be written.<br>• The serial interface IICA0 is in the reset status. |
| 1 | Enables input clock supply.<br>• SFR used by the serial interface IICA0 can be read and written. |

| SAU1EN | Control of serial array unit 1 input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFR used by the serial array unit 1 cannot be written.<br>• The serial array unit 1 is in the reset status. |
| 1 | Enables input clock supply.<br>• SFR used by the serial array unit 1 can be read and written. |

**Caution Be sure to clear the following bits to 0.**
**25-pin products: bits 1, 3, 6**
**32, 48, 64-pin products: bits 1, 6**

**Figure 5-7. Format of Peripheral Enable Register 0 (PER0) (3/3)**

Address: F00F0H After reset: 00H R/W

| Symbol | <7> | 6 | <5> | <4> | <3> | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER0 | RTCEN | 0 | ADCEN | IICA0EN | SAU1EN | SAU0EN | 0 | TAU0EN |

| SAU0EN | Control of serial array unit 0 input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFR used by the serial array unit 0 cannot be written.<br>• The serial array unit 0 is in the reset status. |
| 1 | Enables input clock supply.<br>• SFR used by the serial array unit 0 can be read and written. |

| TAU0EN | Control of timer array unit 0 input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFR used by timer array unit 0 cannot be written.<br>• Timer array unit 0 is in the reset status. |
| 1 | Enables input clock supply.<br>• SFR used by timer array unit 0 can be read and written. |

**Caution Be sure to clear the following bits to 0.**
**25-pin products: bits 1, 3, 6**
**32, 48, 64-pin products: bits 1, 6**

### <R> 5.3.7 Subsystem clock supply mode control register (OSMC)

This register is used to reduce power consumption by stopping unnecessary clock functions.

If the RTCLPC bit is set to 1, power consumption can be reduced, because clock supply to the peripheral functions, except the real-time clock and 12-bit interval timer, is stopped in STOP mode or HALT mode while subsystem clock is <R> selected as CPU clock.

<R> In addition, the OSMC register can be used to select the count clock of the real-time clock and 12-bit interval timer.

The OSMC register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

<R> **Figure 5-8. Format of Subsystem Clock Supply Mode Control Register (OSMC)**

Address: F00F3H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| OSMC | RTCLPC | 0 | 0 | WUTMMCK0 | 0 | 0 | 0 | 0 |

| RTCLPC | Setting in STOP mode or HALT mode while subsystem clock is selected as CPU clock |
|---|---|
| 0 | Enables supply of subsystem clock to peripheral functions<br>(See Tables 18-1, 18-2, and 18-3 for peripheral functions whose operations are enabled.) |
| 1 | Stops supply of subsystem clock to peripheral functions other than real-time clock and 12-bit interval timer. |

<R>

| WUTMMCK0 | Selection of count clock for real-time clock and 12-bit interval timer. |
|---|---|
| 0 | Subsystem clock ($f_{SUB}$) |
| 1 | Low-speed on-chip oscillator clock ($f_{IL}$) |

### 5.3.8 High-speed on-chip oscillator frequency select register (HOCODIV)

The frequency of the high-speed on-chip oscillator which is set by an option byte (000C2H) can be changed by using high-speed on-chip oscillator frequency select register (HOCODIV). However, the selectable frequency depends on the FRQSEL3 bit of the option byte (000C2H).

The HOCODIV register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to the value set by FRQSEL2 to FRQSEL0 of the option byte (000C2H).

**Figure 5-9. Format of High-speed On-chip Oscillator Frequency Select Register (HOCODIV)**

<R> Address: F00A8H After reset: the value set by FRQSEL2 to FRQSEL0 of the option byte (000C2H) R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| HOCODIV | 0 | 0 | 0 | 0 | 0 | HOCODIV2 | HOCODIV1 | HOCODIV0 |

| HOCODIV2 | HOCODIV1 | HOCODIV0 | High-Speed On-Chip Oscillator Clock Frequency | |
|---|---|---|---|---|
| | | | FRQSEL3 Bit is 0 | FRQSEL3 Bit of is 1 |
| 0 | 0 | 0 | 24 MHz | 32 MHz |
| 0 | 0 | 1 | 12 MHz | 16 MHz |
| 0 | 1 | 0 | 6 MHz | 8 MHz |
| 0 | 1 | 1 | 3 MHz | 4 MHz |
| 1 | 0 | 0 | Setting prohibited | 2 MHz |
| 1 | 0 | 1 | Setting prohibited | 1 MHz |
| Other than aboves | | | Setting prohibited | |

**Cautions 1. When changing the frequency by using the HOCODIV register, specify a frequency in the voltage range corresponding to the flash operating mode specified in the option byte (000C2H).**

| Option Byte (000C2H) Value | | Flash Operation Mode | Operating Frequency Range | Operating Voltage Range |
|---|---|---|---|---|
| CMODE1 | CMODE0 | | | |
| 0 | 0 | LV (low-voltage main) mode | 1 to 4 MHz | 1.6 to 3.6 V |
| 1 | 0 | LS (low-speed main) mode | 1 to 8 MHz | 1.8 to 3.6 V |
| 1 | 1 | HS (high-speed main) mode | 1 to 16 MHz | 2.4 to 3.6 V |
| | | | 1 to 32 MHz | 2.7 to 3.6 V |

**2. Change the HOCODIV register value while the high-speed on-chip oscillator clock ($f_{IH}$) is selected as the CPU/peripheral hardware clock ($f_{CLK}$).**

<R> **3. After the frequency is changed using the HOCODIV register, the new frequency is applied after the period below has elapsed.**

- **Operation for up to three clocks at the pre-change frequency**
- **CPU/peripheral hardware clock wait at the post-change frequency for up to three clocks**

### 5.3.9 High-speed on-chip oscillator trimming register (HIOTRM)

This register is used to adjust the accuracy of the high-speed on-chip oscillator.

With self-measurement of the high-speed on-chip oscillator frequency via a timer using high-accuracy external clock input (timer array unit), and so on, the accuracy can be adjusted.

The HIOTRM register can be set by an 8-bit memory manipulation instruction.

**Caution The frequency will vary if the temperature and $V_{DD}$ pin voltage change after accuracy adjustment. When the temperature and $V_{DD}$ voltage change, accuracy adjustment must be executed regularly or before the frequency accuracy is required.**

**Figure 5-10. Format of High-Speed On-Chip Oscillator Trimming Register (HIOTRM)**

Address: F00A0H After reset: Undefined[Note] R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| HIOTRM | 0 | 0 | HIOTRM5 | HIOTRM4 | HIOTRM3 | HIOTRM2 | HIOTRM1 | HIOTRM0 |

| HIOTRM5 | HIOTRM4 | HIOTRM3 | HIOTRM2 | HIOTRM1 | HIOTRM0 | High-speed on-chip oscillator |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | Minimum speed |
| 0 | 0 | 0 | 0 | 0 | 1 | ↑ |
| 0 | 0 | 0 | 0 | 1 | 0 | |
| 0 | 0 | 0 | 0 | 1 | 1 | |
| 0 | 0 | 0 | 1 | 0 | 0 | |
| • | • | • | • | • | • | |
| 1 | 1 | 1 | 1 | 1 | 0 | ↓ |
| 1 | 1 | 1 | 1 | 1 | 1 | Maximum speed |

**Note** The value after reset is the value adjusted at shipment.

**Remarks 1.** The HIOTRM register can be used to adjust the high-speed on-chip oscillator clock to an accuracy within about 0.05%.

**2.** For the usage example of the HIOTRM register, see the application note for RL78 MCU series High-speed On-chip Oscillator (HOCO) Clock Frequency Correction (R01AN0464).

## 5.4 System Clock Oscillator

### 5.4.1 X1 oscillator

The X1 oscillator oscillates with a crystal resonator or ceramic resonator (1 to 20 MHz) connected to the X1 and X2 pins.

An external clock can also be input. In this case, input the clock signal to the EXCLK pin.

To use the X1 oscillator, set bits 7 and 6 (EXCLK, OSCSEL) of the clock operation mode control register (CMC) as follows.

- Crystal or ceramic oscillation: EXCLK, OSCSEL = 0, 1
- External clock input: EXCLK, OSCSEL = 1, 1

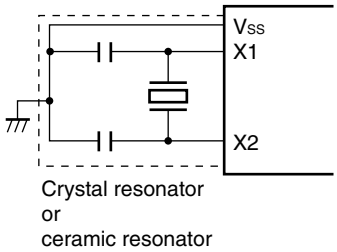
When the X1 oscillator is not used, set the input port mode (EXCLK, OSCSEL = 0, 0).

When the pins are not used as input port pins, either, see **Table 2-3 Connection of Unused Pins (64-pin products)**.

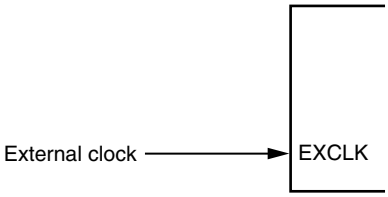
Figure 5-11 shows an example of the external circuit of the X1 oscillator.

**Figure 5-11. Example of External Circuit of X1 Oscillator**

**(a) Crystal or ceramic oscillation**




**(b) External clock**




Cautions are listed on the next page.

### 5.4.2 XT1 oscillator

The XT1 oscillator oscillates with a crystal resonator (32.768 kHz (TYP.)) connected to the XT1 and XT2 pins.

To use the XT1 oscillator, set bit 4 (OSCSELS) of the clock operation mode control register (CMC) to 1.

An external clock can also be input. In this case, input the clock signal to the EXCLKS pin.

To use the XT1 oscillator, set bits 5 and 4 (EXCLKS, OSCSELS) of the clock operation mode control register (CMC) as follows.

- Crystal or ceramic oscillation: EXCLKS, OSCSELS = 0, 1
- External clock input: EXCLKS, OSCSELS = 1, 1

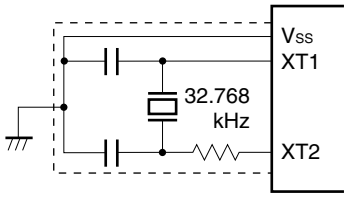
When the XT1 oscillator is not used, set the input port mode (EXCLKS, OSCSELS = 0, 0).

When the pins are not used as input port pins, either, see **Table 2-3 Connection of Unused Pins (64-pin products)**.

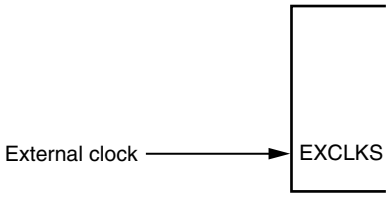
Figure 5-12 shows an example of the external circuit of the XT1 oscillator.

**Figure 5-12. Example of External Circuit of XT1 Oscillator**

**(a) Crystal oscillation**




**(b) External clock**




**Caution When using the X1 oscillator and XT1 oscillator, wire as follows in the area enclosed by the broken lines in the Figures 5-11 and 5-12 to avoid an adverse effect from wiring capacitance.**

- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with the other signal lines. Do not route the wiring near a signal line through which a high fluctuating current flows.**
- **Always make the ground point of the oscillator capacitor the same potential as Vss. Do not ground the capacitor to a ground pattern through which a high current flows.**
- **Do not fetch signals from the oscillator.**

**The XT1 oscillator is a circuit with low amplification in order to achieve low-power consumption. Note the following points when designing the circuit.**

- **Pins and circuit boards include parasitic capacitance. Therefore, perform oscillation evaluation using a circuit board to be actually used and confirm that there are no problems.**

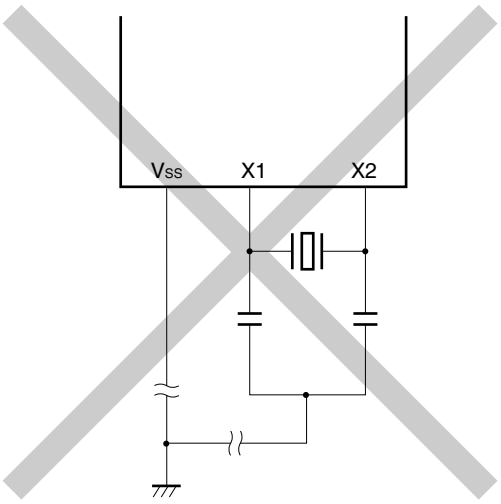
<R>

- **Before using the ultra-low power consumption oscillation (AMPHS1, AMPHS0 = 1, 0) as the mode of the XT1 oscillator, evaluate the resonators described in 5.7 Resonator and Oscillator Constants.**
- **Make the wiring between the XT1 and XT2 pins and the resonators as short as possible, and minimize the parasitic capacitance and wiring resistance. Note this particularly when the ultra-low power consumption oscillation (AMPHS1, AMPHS0 = 1, 0) is selected.**
- **Configure the circuit of the circuit board, using material with little wiring resistance.**
- **Place a ground pattern that has the same potential as Vss as much as possible near the XT1 oscillator.**
- **Be sure that the signal lines between the XT1 and XT2 pins, and the resonators do not cross with the other signal lines. Do not route the wiring near a signal line through which a high fluctuating current flows.**
- **The impedance between the XT1 and XT2 pins may drop and oscillation may be disturbed due to moisture absorption of the circuit board in a high-humidity environment or dew condensation on the board. When using the circuit board in such an environment, take measures to damp-proof the circuit board, such as by coating.**
- **When coating the circuit board, use material that does not cause capacitance or leakage between the XT1 and XT2 pins.**

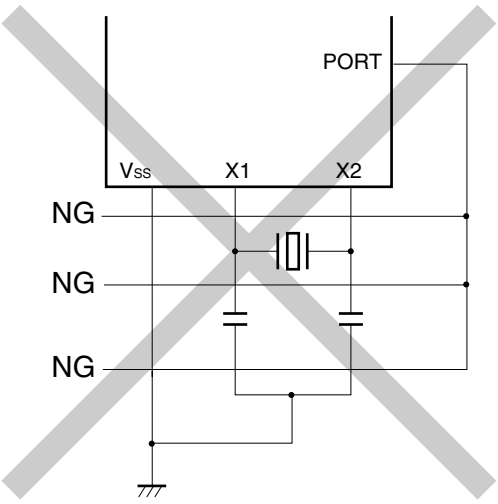
Figure 5-13 shows examples of incorrect resonator connection.

**Figure 5-13. Examples of Incorrect Resonator Connection (1/2)**

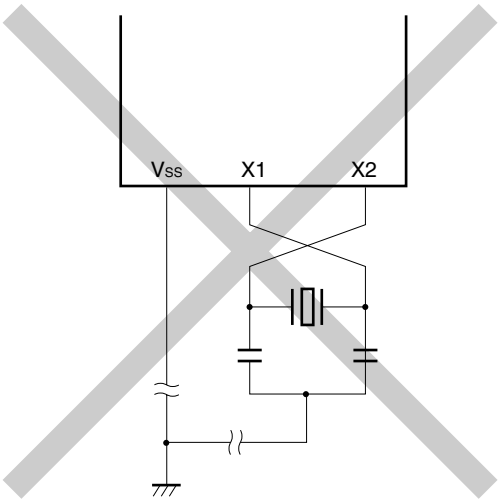
**(a) Too long wiring**



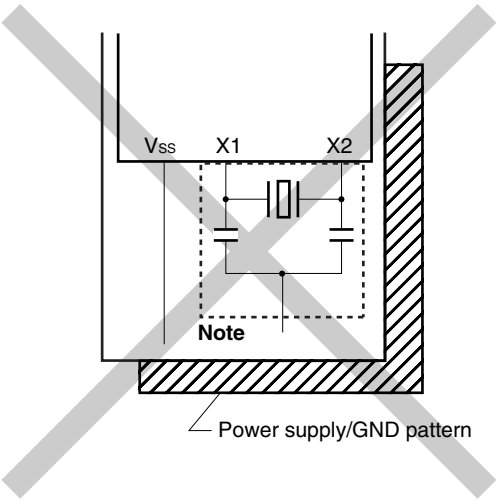

**(b) Crossed signal line**




**(c) The X1 and X2 signal line wires cross.**




**(d) A power supply/GND pattern exists under the X1 and X2 wires.**

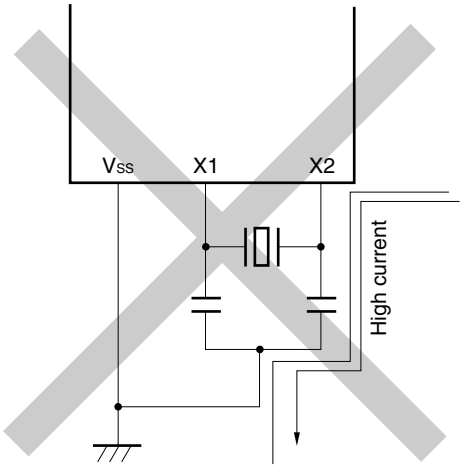



**Note** Do not place a power supply/GND pattern under the wiring section (section indicated by a broken line in the figure) of the X1 and X2 pins and the resonators in a multi-layer board or double-sided board.
Do not configure a layout that will cause capacitance elements and affect the oscillation characteristics.

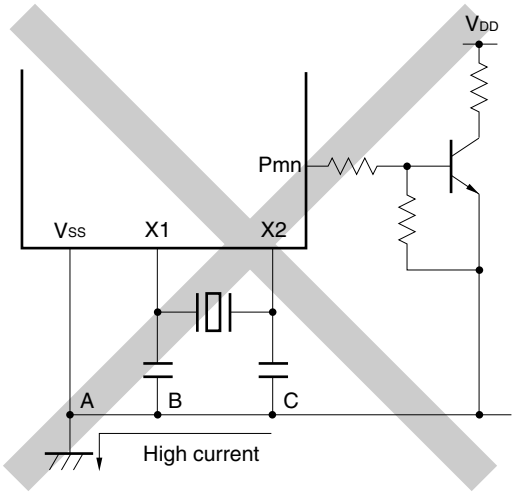
**Remark** When using the subsystem clock, replace X1 and X2 with XT1 and XT2, respectively. Also, insert resistors in series on the XT2 side.

**Figure 5-13. Examples of Incorrect Resonator Connection (2/2)**

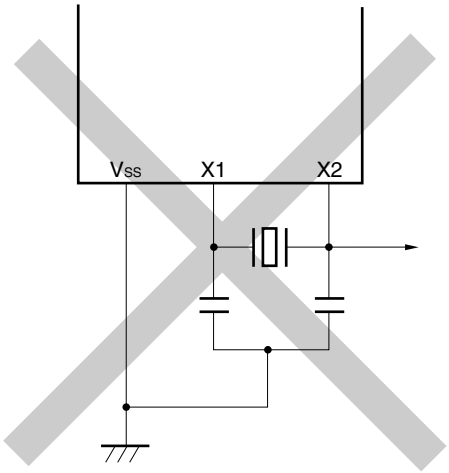
**(e) Wiring near high alternating current**




**(f) Current flowing through ground line of oscillator (potential at points A, B, and C fluctuates)**




**(g) Signals are fetched**




**Caution When X2 and XT1 are wired in parallel, the crosstalk noise of X2 may increase with XT1, resulting in malfunctioning.**

**Remark** When using the subsystem clock, replace X1 and X2 with XT1 and XT2, respectively. Also, insert resistors in series on the XT2 side.

### 5.4.3 High-speed on-chip oscillator

The high-speed on-chip oscillator is incorporated in the RL78/G1A. The frequency can be selected from among 32, 24, 16, 12, 8, 6, 4, 3, 2, or 1 MHz by using the option byte (000C2H). Oscillation can be controlled by bit 0 (HIOSTOP) of the clock operation status control register (CSC). The high-speed on-chip oscillator automatically starts oscillating after reset release.

### 5.4.4 Low-speed on-chip oscillator

The low-speed on-chip oscillator is incorporated in the RL78/G1A.

The low-speed on-chip oscillator clock is used only as the watchdog timer, real-time clock, and 12-bit interval timer clock. The low-speed on-chip oscillator clock cannot be used as the CPU clock.

<R> This clock operates when bit 4 (WDTON) of the option byte (000C0H), bit 4 (WUTMMCK0) of the subsystem clock supply mode control register (OSMC), or both are set to 1.

Unless the watchdog timer is stopped and WUTMMCK0 is a value other than zero, oscillation of the low-speed on-chip
<R> oscillator continues. Note that only when the watchdog timer is operating and the WUTMMCK0 bit is 0, oscillation of the low-speed on-chip oscillator will stop while the WDSTBYON bit is 0 and operation is in the HALT, STOP, or SNOOZE mode. While the watchdog timer operates, the low-speed on-chip oscillator clock does not stop even if the program freezes.

## 5.5 Clock Generator Operation

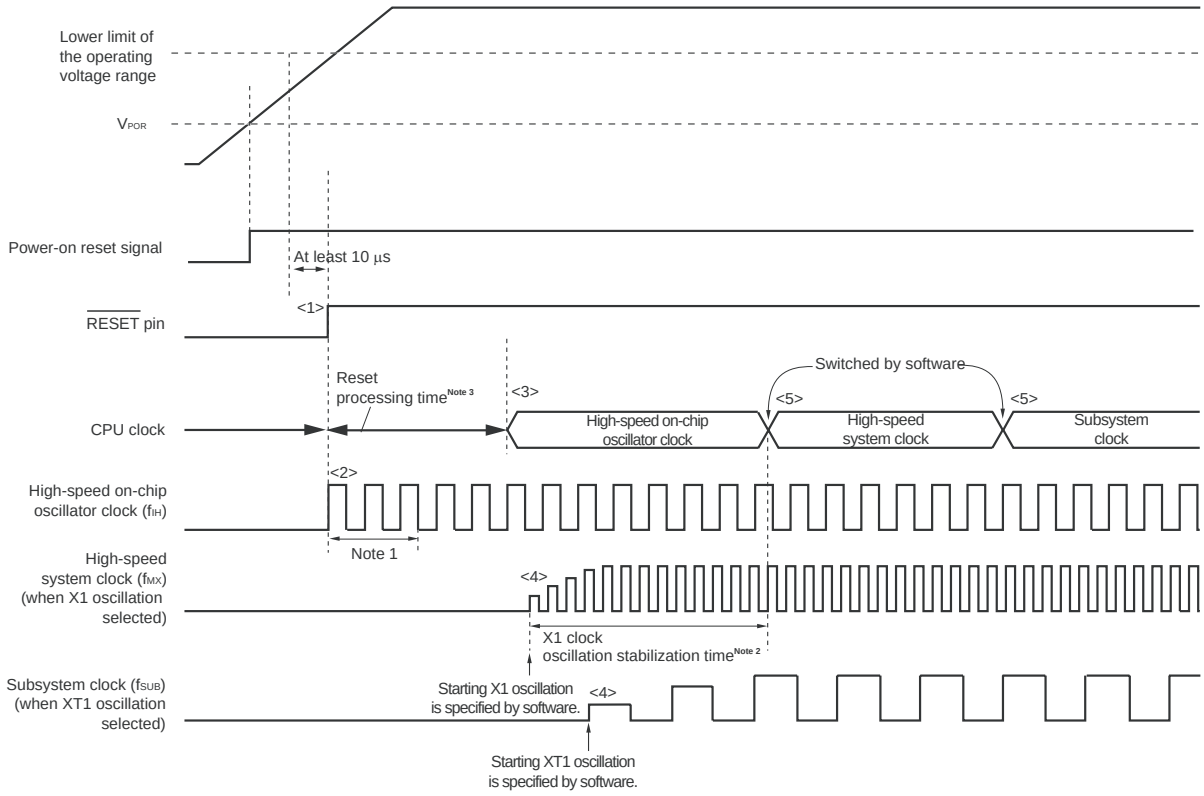
The clock generator generates the following clocks and controls the operation modes of the CPU, such as standby mode (see **Figure 5-1**).

- Main system clock $f_{MAIN}$
  - High-speed system clock $f_{MX}$
    X1 clock $f_X$
    External main system clock $f_{EX}$
  - High-speed on-chip oscillator clock $f_{IH}$
- Subsystem clock $f_{SUB}$
  - XT1 clock $f_{XT}$
  - External subsystem clock $f_{EXS}$
- Low-speed on-chip oscillator clock $f_{IL}$
- CPU/peripheral hardware clock $f_{CLK}$

The CPU starts operation when the high-speed on-chip oscillator starts outputting after a reset release in the RL78/G1A.

When the power supply voltage is turned on, the clock generator operation is shown in Figure 5-14.

**Figure 5-14. Clock Generator Operation When Power Supply Voltage Is Turned On**




<1> When the power is turned on, an internal reset signal is generated by the power-on-reset (POR) circuit. Note that the reset state is maintained after a reset by the voltage detection circuit or an external reset until the voltage reaches the range of operating voltage described in **29.4 AC Characteristics** and **30.4 AC Characteristics** (the above figure is an example when the external reset is in use).

<2> When the reset is released, the high-speed on-chip oscillator automatically starts oscillation.

<3> The CPU starts operation on the high-speed on-chip oscillator clock after waiting for the voltage to stabilize and a reset processing have been performed after reset release.

<4> Set the start of oscillation of the X1 or XT1 clock via software (see **5.6.2 Example of setting X1 oscillation clock** and **5.6.3 Example of setting XT1 oscillation clock**).

<5> When switching the CPU clock to the X1 or XT1 clock, wait for the clock oscillation to stabilize, and then set switching via software (see **5.6.2 Example of setting X1 oscillation clock** and **5.6.3 Example of setting XT1 oscillation clock**).

**Notes 1.** The internal reset processing time includes the oscillation accuracy stabilization time of the high-speed on-chip oscillator clock.

**2.** When releasing a reset, confirm the oscillation stabilization time for the X1 clock using the oscillation stabilization time counter status register (OSTC).

3. For the reset processing time, see **CHAPTER 20 POWER-ON-RESET CIRCUIT.**

**Caution It is not necessary to wait for the oscillation stabilization time when an external clock input from the EXCLK pin is used.**

## 5.6 Controlling Clock

### 5.6.1 Example of setting high-speed on-chip oscillator

After a reset release, the CPU/peripheral hardware clock ($f_{CLK}$) always starts operating with the high-speed on-chip oscillator clock. The frequency of the high-speed on-chip oscillator can be selected from 32, 24, 16, 12, 8, 6, 4, 3, 2, and 1 MHz by using FRQSEL0 to FRQSEL3 of the option byte (000C2H).

[Option byte setting]

Address: 000C2H

Option byte (000C2H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CMODE1 | CMODE0 | | | FRQSEL3 | FRQSEL2 | FRQSEL1 | FRQSEL0 |
| 0/1 | 0/1 | 1 | 0 | 0/1 | 0/1 | 0/1 | 0/1 |

| CMODE1 | CMODE0 | Setting of flash operation mode | |
|---|---|---|---|
| 0 | 0 | LV (low voltage main) mode | $V_{DD}$ = 1.6 V to 3.6 V @ 1 MHz to 4 MHz |
| 1 | 0 | LS (low speed main) mode | $V_{DD}$ = 1.8 V to 3.6 V @ 1 MHz to 8 MHz |
| 1 | 1 | HS (high speed main) mode | $V_{DD}$ = 2.4 V to 3.6 V @ 1 MHz to 16 MHz<br>$V_{DD}$ = 2.7 V to 3.6 V @ 1 MHz to 32 MHz |
| Other than above | | Setting prohibited | |

| FRQSEL3 | FRQSEL2 | FRQSEL1 | FRQSEL0 | Frequency of the high-speed on-chip oscillator |
|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 32 MHz |
| 0 | 0 | 0 | 0 | 24 MHz |
| 1 | 0 | 0 | 1 | 16 MHz |
| 0 | 0 | 0 | 1 | 12 MHz |
| 1 | 0 | 1 | 0 | 8 MHz |
| 0 | 0 | 1 | 0 | 6 MHz |
| 1 | 0 | 1 | 1 | 4 MHz |
| 0 | 0 | 1 | 1 | 3 MHz |
| 1 | 1 | 0 | 0 | 2 MHz |
| 1 | 1 | 0 | 1 | 1 MHz |
| Other than above | | | | Setting prohibited |

[High-speed on-chip oscillator frequency select register (HOCODIV) setting]

Address: F00A8H

HOCODIV

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | HOCODIV2 | HOCODIV1 | HOCODIV0 |

| HOCODIV2 | HOCODIV1 | HOCODIV0 | Selection of high-speed on-chip oscillator clock frequency | |
|---|---|---|---|---|
| | | | FRQSEL3 Bit is 0 | FRQSEL3 Bit of is 1 |
| 0 | 0 | 0 | 24 MHz | 32 MHz |
| 0 | 0 | 1 | 12 MHz | 16 MHz |
| 0 | 1 | 0 | 6 MHz | 8 MHz |
| 0 | 1 | 1 | 3 MHz | 4 MHz |
| 1 | 0 | 0 | Setting prohibited | 2 MHz |
| 1 | 0 | 1 | Setting prohibited | 1 MHz |
| Other than above | | | Setting prohibited | |

### 5.6.2 Example of setting X1 oscillation clock

After a reset release, the CPU/peripheral hardware clock ($f_{CLK}$) always starts operating with the high-speed on-chip oscillator clock. To subsequently change the clock to the X1 oscillation clock, set the oscillator and start oscillation by using the oscillation stabilization time select register (OSTS) and clock operation mode control register (CMC) and clock operation status control register (CSC) and wait for oscillation to stabilize by using the oscillation stabilization time counter status register (OSTC). After the oscillation stabilizes, set the X1 oscillation clock to $f_{CLK}$ by using the system clock control register (CKC).

[Register settings] Set the register in the order of <1> to <5> below.

<1> Set (1) the OSCSEL bit of the CMC register, except for the cases where $f_X$ > 10 MHz, in such cases set (1) the AMPH bit, to operate the X1 oscillator.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CMC | EXCLK | OSCSEL | EXCLKS | OSCSELS | | AMPHS1 | AMPHS0 | AMPH |
| | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0/1 |

<2> Using the OSTS register, select the oscillation stabilization time of the X1 oscillator at releasing of the STOP mode.
Example: Setting values when a wait of at least 102 $\mu$s is set based on a 10 MHz resonator.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| OSTS | | | | | | OSTS2 | OSTS1 | OSTS0 |
| | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |

<3> Clear (0) the MSTOP bit of the CSC register to start oscillating the X1 oscillator.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CSC | MSTOP | XTSTOP | | | | | | HIOSTOP |
| | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

<4> Use the OSTC register to wait for oscillation of the X1 oscillator to stabilize.
Example: Wait until the bits reach the following values when a wait of at least 102 $\mu$s is set based on a 10 MHz resonator.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| OSTC | MOST8 | MOST9 | MOST10 | MOST11 | MOST13 | MOST15 | MOST17 | MOST18 |
| | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |

<5> Use the MCM0 bit of the CKC register to specify the X1 oscillation clock as the CPU/peripheral hardware clock.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CKC | CLS | CSS | MCS | MCM0 | | | | |
| | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

### 5.6.3 Example of setting XT1 oscillation clock

After a reset release, the CPU/peripheral hardware clock ($f_{CLK}$) always starts operating with the high-speed on-chip oscillator clock. To subsequently change the clock to the XT1 oscillation clock, set the oscillator and start oscillation by
<R> using the subsystem clock supply mode control register (OSMC), clock operation mode control register (CMC), and clock operation status control register (CSC), set the XT1 oscillation clock to $f_{CLK}$ by using the system clock control register (CKC).

[Register settings] Set the register in the order of <1> to <5> below.

<1> To run only the real-time clock and 12-bit interval timer on the subsystem clock (ultra-low current consumption) when in the STOP mode or sub-HALT mode, set the RTCLPC bit to 1.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| OSMC | RTCLPC | | | WUTMMCK0 | | | | |
| | 0/1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

<2> Set (1) the OSCSELS bit of the CMC register to operate the XT1 oscillator.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CMC | EXCLK | OSCSEL | EXCLKS | OSCSELS | | AMPHS1 | AMPHS0 | AMPH |
| | 0 | 0 | 0 | 1 | 0 | 0/1 | 0/1 | 0 |

AMPHS0 and AMPHS1 bits: These bits are used to specify the oscillation mode of the XT1 oscillator.

<3> Clear (0) the XTSTOP bit of the CSC register to start oscillating the XT1 oscillator.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CSC | MSTOP | XTSTOP | | | | | | HIOSTOP |
| | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

<4> Use the timer function or another function to wait for oscillation of the subsystem clock to stabilize by using software.

<5> Use the CSS bit of the CKC register to specify the XT1 oscillation clock as the CPU/peripheral hardware clock.

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CKC | CLS | CSS | MCS | MCM0 | | | | |
| | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

### 5.6.4 CPU clock status transition diagram

Figure 5-15 shows the CPU clock status transition diagram of this product.

**Figure 5-15. CPU Clock Status Transition Diagram**

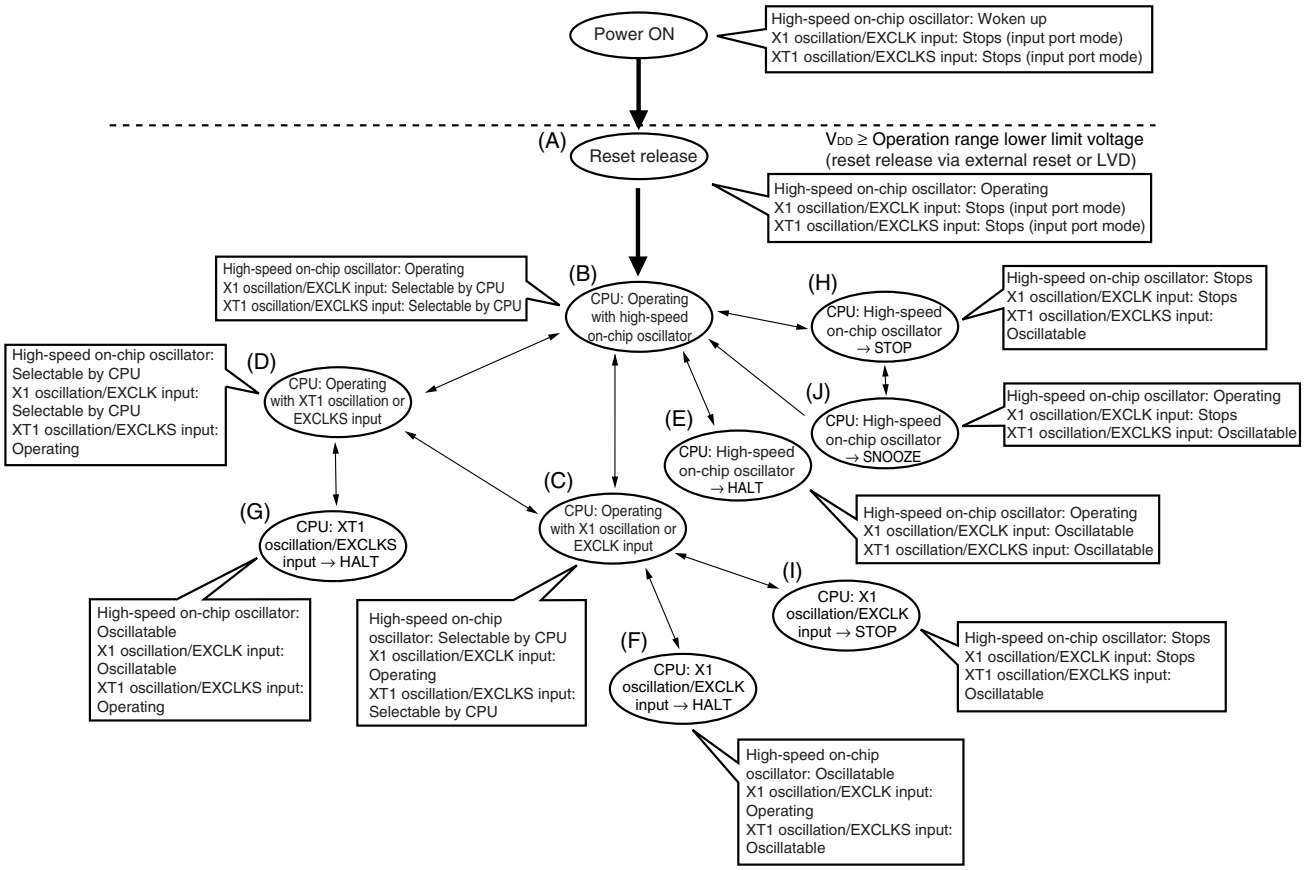


Table 5-3 shows transition of the CPU clock and examples of setting the SFR registers.

**Table 5-3. CPU Clock Transition and SFR Register Setting Examples (1/5)**

**(1) CPU operating with high-speed on-chip oscillator clock (B) after reset release (A)**

| Status Transition | SFR Register Setting |
|---|---|
| (A) → (B) | SFR registers do not have to be set (default status after reset release). |

**(2) CPU operating with high-speed system clock (C) after reset release (A)**

(The CPU operates with the high-speed on-chip oscillator clock immediately after a reset release (B).)

(Setting sequence of SFR registers) →

| Setting Flag of SFR Register / Status Transition | CMC Register Note | | | OSTS Register | CSC Register | OSTC Register | CKC Register |
|---|---|---|---|---|---|---|---|
| | EXCLK | OSCSEL | AMPH | | MSTOP | | MCM0 |
| (A) → (B) → (C) (X1 clock: 1 MHz ≤ fx ≤ 10 MHz) | 0 | 1 | 0 | **Note 2** | 0 | Must be checked | 1 |
| (A) → (B) → (C) (X1 clock: 10 MHz < fx ≤ 20 MHz) | 0 | 1 | 1 | **Note 2** | 0 | Must be checked | 1 |
| (A) → (B) → (C) (external main clock) | 1 | 1 | × | **Note 2** | 0 | Must not be checked | 1 |

**Notes 1.** The clock operation mode control register (CMC) can be written only once by an 8-bit memory manipulation instruction after reset release.

**2.** Set the oscillation stabilization time as follows.

- Desired the oscillation stabilization time counter status register (OSTC) oscillation stabilization time ≤ Oscillation stabilization time set by the oscillation stabilization time select register (OSTS)

**Caution Set the clock after the supply voltage has reached the operable voltage of the clock to be set (see CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C), CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to + 105°C).**

**(3) CPU operating with subsystem clock (D) after reset release (A)**

(The CPU operates with the high-speed on-chip oscillator clock immediately after a reset release (B).)

(Setting sequence of SFR registers) →

| Setting Flag of SFR Register / Status Transition | CMC Register Note | | | | CSC Register | Waiting for Oscillation Stabilization | CKC Register |
|---|---|---|---|---|---|---|---|
| | EXCLKS | OSCSELS | AMPHS1 | AMPHS0 | XTSTOP | | CSS |
| (A) → (B) → (D) (XT1 clock) | 0 | 1 | 0/1 | 0/1 | 0 | Necessary | 1 |
| (A) → (B) → (D) (external sub clock) | 1 | 1 | × | × | 0 | Necessary | 1 |

**Note** The clock operation mode control register (CMC) can be written only once by an 8-bit memory manipulation instruction after reset release.

**Remarks 1.** ×: don't care

**2.** (A) to (J) in Table 5-3 correspond to (A) to (J) in Figure 5-15.

**Table 5-3. CPU Clock Transition and SFR Register Setting Examples (2/5)**

**(4) CPU clock changing from high-speed on-chip oscillator clock (B) to high-speed system clock (C)**

(Setting sequence of SFR registers) →

| Setting Flag of SFR Register / Status Transition | CMC Register[Note 1] | | | OSTS Register | CSC Register | OSTC Register | CKC Register |
|---|---|---|---|---|---|---|---|
| | EXCLK | OSCSEL | AMPH | | MSTOP | | MCM0 |
| (B) → (C) (X1 clock: 1 MHz ≤ fX ≤ 10 MHz) | 0 | 1 | 0 | **Note 2** | 0 | Must be checked | 1 |
| (B) → (C) (X1 clock: 10 MHz < fX ≤ 20 MHz) | 0 | 1 | 1 | **Note 2** | 0 | Must be checked | 1 |
| (B) → (C) (external main clock) | 1 | 1 | × | **Note 2** | 0 | Must not be checked | 1 |

Unnecessary if these registers are already set

Unnecessary if the CPU is operating with the high-speed system clock

**Notes 1.** The clock operation mode control register (CMC) can be written only once by an 8-bit memory manipulation instruction after reset release.
This setting is not necessary if it has already been set.

**2.** Set the oscillation stabilization time as follows.

- Desired the oscillation stabilization time counter status register (OSTC) oscillation stabilization time ≤ Oscillation stabilization time set by the oscillation stabilization time select register (OSTS)

**Caution Set the clock after the supply voltage has reached the operable voltage of the clock to be set (see CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C), CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to + 105°C).**

**(5) CPU clock changing from high-speed on-chip oscillator clock (B) to subsystem clock (D)**

<R> (Setting sequence of SFR registers) →

| Setting Flag of SFR Register / Status Transition | CMC Register[Note] | | | CSC Register | Waiting for Oscillation Stabilization | CKC Register |
|---|---|---|---|---|---|---|
| | EXCLKS | OSCSELS | AMPHS1, AMPHS0 | XTSTOP | | CSS |
| (B) → (D) (XT1 clock) | 0 | 1 | 00: Low power consumption oscillation<br>01: Normal oscillation<br>10: Ultra-low power consumption oscillation | 0 | Necessary | 1 |
| (B) → (D) (external sub clock) | 1 | 1 | × | 0 | Necessary | 1 |

Unnecessary if these registers are already set

Unnecessary if the CPU is operating with the subsystem clock

**Note** The clock operation mode control register (CMC) can be written only once by an 8-bit memory manipulation instruction after reset release.

<R> This setting is not necessary if it has already been set.

**Remarks 1.** ×: don't care

**2.** (A) to (J) in Table 5-3 correspond to (A) to (J) in Figure 5-15.

**Table 5-3. CPU Clock Transition and SFR Register Setting Examples (3/5)**

### (6) CPU clock changing from high-speed system clock (C) to high-speed on-chip oscillator clock (B)

<R>

(Setting sequence of SFR registers) →

| Setting Flag of SFR Register / Status Transition | CSC Register | Oscillation accuracy stabilization time | CKC Register |
|---|---|---|---|
| | HIOSTOP | | MCM0 |
| (C) → (B) | 0 | 18 μs to 65 μs | 0 |

Unnecessary if the CPU is operating with the high-speed on-chip oscillator clock

**Remark** The oscillation accuracy stabilization time changes according to the temperature conditions and the STOP mode period.

### (7) CPU clock changing from high-speed system clock (C) to subsystem clock (D)

(Setting sequence of SFR registers) →

| Setting Flag of SFR Register / Status Transition | CSC Register | Waiting for Oscillation Stabilization | CKC Register |
|---|---|---|---|
| | XTSTOP | | CSS |
| (C) → (D) | 0 | Necessary | 1 |

Unnecessary if the CPU is operating with the subsystem clock

<R>

### (8) CPU clock changing from subsystem clock (D) to high-speed on-chip oscillator clock (B)

(Setting sequence of SFR registers) →

| Setting Flag of SFR Register / Status Transition | CSC Register | Oscillation accuracy stabilization time | CKC Register |
|---|---|---|---|
| | HIOSTOP | | CSS |
| (D) → (B) | 0 | 18 μs to 65 μs | 0 |

Unnecessary if the CPU is operating with the high-speed on-chip oscillator clock

**Remarks 1.** (A) to (J) in Table 5-3 correspond to (A) to (J) in Figure 5-15.

**2.** The oscillation accuracy stabilization time changes according to the temperature conditions and the STOP mode period.

**Table 5-3. CPU Clock Transition and SFR Register Setting Examples (4/5)**

**(9) CPU clock changing from subsystem clock (D) to high-speed system clock (C)**

<R> (Setting sequence of SFR registers) ⟶

| Setting Flag of SFR Register / Status Transition | OSTS Register | CSC Register | OSTC Register | CKC Register |
|---|---|---|---|---|
| | | MSTOP | | CSS |
| (D) → (C) (X1 clock: 1 MHz ≤ fx ≤ 10 MHz) | **Note** | 0 | Must be checked | 0 |
| (D) → (C) (X1 clock: 10 MHz < fx ≤ 20 MHz) | **Note** | 0 | Must be checked | 0 |
| (D) → (C) (external main clock) | **Note** | 0 | Must not be checked | 0 |

Unnecessary if the CPU is operating with the high-speed system clock

**Note** Set the oscillation stabilization time as follows.

- Desired the oscillation stabilization time counter status register (OSTC) oscillation stabilization time ≤ Oscillation stabilization time set by the oscillation stabilization time select register (OSTS)

**Caution Set the clock after the supply voltage has reached the operable voltage of the clock to be set (see CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C), CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to + 105°C).**

**(10) • HALT mode (E) set while CPU is operating with high-speed on-chip oscillator clock (B)**

**• HALT mode (F) set while CPU is operating with high-speed system clock (C)**

**• HALT mode (G) set while CPU is operating with subsystem clock (D)**

| Status Transition | Setting |
|---|---|
| (B) → (E)<br>(C) → (F)<br>(D) → (G) | Executing HALT instruction |

**Remark** (A) to (J) in Table 5-3 correspond to (A) to (J) in Figure 5-15.

**Table 5-3. CPU Clock Transition and SFR Register Setting Examples (5/5)**

**(11) • STOP mode (H) set while CPU is operating with high-speed on-chip oscillator clock (B)**
**• STOP mode (I) set while CPU is operating with high-speed system clock (C)**

<R> (Setting sequence) ——————————————————————▶

| Status Transition | | Setting | | |
|---|---|---|---|---|
| (B) → (H) | | Stopping peripheral functions that are disabled in STOP mode | – | Executing STOP instruction |
| (C) → (I) | In X1 oscillation | | Sets the OSTS register | |
| | External main system clock | | – | |

**(12) CPU changing from STOP mode (H) to SNOOZE mode (J)**

For details about the setting for switching from the STOP mode to the SNOOZE mode, see **11.8 SNOOZE Mode Function**, **12.5.7 SNOOZE mode function** and **12.6.3 SNOOZE mode function**.

**Remark** (A) to (J) in Table 5-3 correspond to (A) to (J) in Figure 5-15.

### 5.6.5 Condition before changing CPU clock and processing after changing CPU clock

Condition before changing the CPU clock and processing after changing the CPU clock are shown below.

**Table 5-4. Changing CPU Clock (1/2)**

| | CPU Clock | | Condition Before Change | Processing After Change |
|---|---|---|---|---|
| | Before Change | After Change | | |
| | High-speed on-chip oscillator clock | X1 clock | Stabilization of X1 oscillation<br>• OSCSEL = 1, EXCLK = 0, MSTOP = 0<br>• After elapse of oscillation stabilization time | Operating current can be reduced by stopping high-speed on-chip oscillator (HIOSTOP = 1). |
| | | External main system clock | Enabling input of external clock from the EXCLK pin<br>• OSCSEL = 1, EXCLK = 1, MSTOP = 0 | |
| | | XT1 clock | Stabilization of XT1 oscillation<br>• OSCSELS = 1, EXCLKS = 0, XTSTOP = 0<br>• After elapse of oscillation stabilization time | |
| | | External subsystem clock | Enabling input of external clock from the EXCLKS pin<br>• OSCSELS = 1, EXCLKS = 1, XTSTOP = 0 | |
| <R> | X1 clock | High-speed on-chip oscillator clock | Enabling oscillation of high-speed on-chip oscillator<br>• HIOSTOP = 0<br>• After elapse of oscillation accuracy stabilization time | X1 oscillation can be stopped (MSTOP = 1). |
| | | External main system clock | Transition not possible | – |
| | | XT1 clock | Stabilization of XT1 oscillation<br>• OSCSELS = 1, EXCLKS = 0, XTSTOP = 0<br>• After elapse of oscillation stabilization time | X1 oscillation can be stopped (MSTOP = 1). |
| | | External subsystem clock | Enabling input of external clock from the EXCLKS pin<br>• OSCSELS = 1, EXCLKS = 1, XTSTOP = 0 | X1 oscillation can be stopped (MSTOP = 1). |
| <R> | External main system clock | High-speed on-chip oscillator clock | Enabling oscillation of high-speed on-chip oscillator<br>• HIOSTOP = 0<br>• After elapse of oscillation accuracy stabilization time | External main system clock input can be disabled (MSTOP = 1). |
| | | X1 clock | Transition not possible | – |
| | | XT1 clock | Stabilization of XT1 oscillation<br>• OSCSELS = 1, EXCLKS = 0, XTSTOP = 0<br>• After elapse of oscillation stabilization time | External main system clock input can be disabled (MSTOP = 1). |
| | | External subsystem clock | Enabling input of external clock from the EXCLKS pin<br>• OSCSELS = 1, EXCLKS = 1, XTSTOP = 0 | External main system clock input can be disabled (MSTOP = 1). |

**Table 5-5. Changing CPU Clock (2/2)**

| CPU Clock | | Condition Before Change | Processing After Change |
|---|---|---|---|
| Before Change | After Change | | |
| XT1 clock | High-speed on-chip oscillator clock | Oscillation of high-speed on-chip oscillator and selection of high-speed on-chip oscillator clock as main system clock<br>• HIOSTOP = 0, MCS = 0 | XT1 oscillation can be stopped (XTSTOP = 1) |
| | X1 clock | Stabilization of X1 oscillation and selection of high-speed system clock as main system clock<br>• OSCSEL = 1, EXCLK = 0, MSTOP = 0<br>• After elapse of oscillation stabilization time<br>• MCS = 1 | |
| | External main system clock | Enabling input of external clock from the EXCLK pin and selection of high-speed system clock as main system clock<br>• OSCSEL = 1, EXCLK = 1, MSTOP = 0<br>• MCS = 1 | |
| | External subsystem clock | Transition not possible | – |
| External subsystem clock | High-speed on-chip oscillator clock | Oscillation of high-speed on-chip oscillator and selection of high-speed on-chip oscillator clock as main system clock<br>• HIOSTOP = 0, MCS = 0 | External subsystem clock input can be disabled (XTSTOP = 1). |
| | X1 clock | Stabilization of X1 oscillation and selection of high-speed system clock as main system clock<br>• OSCSEL = 1, EXCLK = 0, MSTOP = 0<br>• After elapse of oscillation stabilization time<br>• MCS = 1 | |
| | External main system clock | Enabling input of external clock from the EXCLK pin and selection of high-speed system clock as main system clock<br>• OSCSEL = 1, EXCLK = 1, MSTOP = 0<br>• MCS = 1 | |
| | XT1 clock | Transition not possible | – |

### 5.6.6 Time required for switchover of CPU clock and system clock

By setting bits 4 and 6 (MCM0, CSS) of the system clock control register (CKC), the CPU clock can be switched (between the main system clock and the subsystem clock), and main system clock can be switched (between the high-speed on-chip oscillator clock and the high-speed system clock).

The actual switchover operation is not performed immediately after rewriting to the CKC register; operation continues on the pre-switchover clock for several clocks (see Table 5-5 to Table 5-7).

Whether the CPU is operating on the main system clock or the subsystem clock can be ascertained using bit 7 (CLS) of the CKC register. Whether the main system clock is operating on the high-speed system clock or high-speed on-chip oscillator clock can be ascertained using bit 5 (MCS) of the CKC register.

When the CPU clock is switched, the peripheral hardware clock is also switched.

**Table 5-5. Maximum Time Required for System Clock Switchover**

| Clock A | Switching directions | Clock B | Remark |
|---|---|---|---|
| $f_{IH}$ | ←→ | $f_{MX}$ | See Table 5-6 |
| $f_{MAIN}$ | ←→ | $f_{SUB}$ | See Table 5-7 |

**Table 5-6. Maximum Number of Clocks Required for $f_{IH}$ ↔ $f_{MX}$**

| Set Value Before Switchover | | Set Value After Switchover | |
|---|---|---|---|
| MCM0 | | MCM0 | |
| | | 0 ($f_{MAIN} = f_{IH}$) | 1 ($f_{MAIN} = f_{MX}$) |
| 0 ($f_{MAIN} = f_{IH}$) | $f_{MX} \geq f_{IH}$ | | 2 clock |
| | $f_{MX} < f_{IH}$ | | $2f_{IH}/f_{MX}$ clock |
| 1 ($f_{MAIN} = f_{MX}$) | $f_{MX} \geq f_{IH}$ | $2f_{MX}/f_{IH}$ clock | |
| | $f_{MX} < f_{IH}$ | 2 clock | |

**Table 5-7. Maximum Number of Clocks Required for $f_{MAIN}$ ↔ $f_{SUB}$**

| Set Value Before Switchover | Set Value After Switchover | |
|---|---|---|
| CSS | CSS | |
| | 0 ($f_{CLK} = f_{MAIN}$) | 1 ($f_{CLK} = f_{SUB}$) |
| 0 ($f_{CLK} = f_{MAIN}$) | | $1 + 2f_{MAIN}/f_{SUB}$ clock |
| 1 ($f_{CLK} = f_{SUB}$) | 3 clock | |

**Remarks 1.** The number of clocks listed in Table 5-6 and Table 5-7 is the number of CPU clocks before switchover.

**2.** Calculate the number of clocks in **Table 5-6** and **Table 5-7** by removing the decimal portion.

**Example** When switching the main system clock from the high-speed system clock to the high-speed on-chip oscillator clock (@ oscillation with $f_{IH}$ = 8 MHz, $f_{MX}$ = 10 MHz)

$2f_{MX}/f_{IH} = 2\ (10/8) = 2.5 \rightarrow 3$ clocks

### 5.6.7 Conditions before clock oscillation is stopped

The following lists the register flag settings for stopping the clock oscillation (disabling external clock input) and conditions before the clock oscillation is stopped.

**Table 5-7. Conditions Before the Clock Oscillation Is Stopped and Flag Settings**

| Clock | Conditions Before Clock Oscillation Is Stopped (External Clock Input Disabled) | Flag Settings of SFR Register |
|---|---|---|
| High-speed on-chip oscillator clock | MCS = 1 or CLS = 1<br>(The CPU is operating on a clock other than the high-speed on-chip oscillator clock.) | HIOSTOP = 1 |
| X1 clock<br>External main system clock | MCS = 0 or CLS = 1<br>(The CPU is operating on a clock other than the high-speed system clock.) | MSTOP = 1 |
| XT1 clock<br>External subsystem clock | CLS = 0<br>(The CPU is operating on a clock other than the subsystem clock.) | XTSTOP = 1 |

## 5.7 Resonator and Oscillator Constants

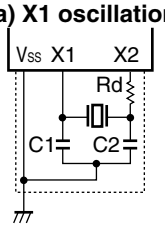
The resonators for which the operation is verified and their oscillator constants are shown below.

**Cautions 1. The oscillator constants shown above are reference values based on evaluation in a specific environment by the resonator manufacturer. Be sure to apply to the resonator manufacturer for evaluation on the actual circuit before using these constants for your application.**
**Also apply to the resonator manufacturer for re-evaluation on the actual circuit if you have changed the make of the microcontroller or the board.**

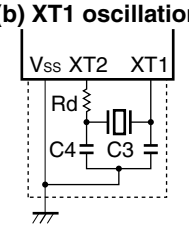
**2. The oscillation voltage and oscillation frequency only indicate the oscillator characteristic. Use the RL78 microcontroller so that the internal operation conditions are within the specifications of the DC and AC characteristics.**

<R> **Figure 5-16. External Oscillation Circuit Example**

**(a) X1 oscillation**




**(b) XT1 oscillation**



**(1) X1 oscillation:** As of February, 2013 (1/3)

| Manufacturer | Resonator | Part Number[Note 1] | SMD/ Lead | Frequency (MHz) | Flash operation mode[Note 2] | Recommended Circuit Constants[Note 3] (reference) | | | Oscillation Voltage Range (V) | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | C1 (pF) | C2 (pF) | Rd (kΩ) | MIN. | MAX. |
| Murata Manufacturing Co., Ltd.[Note 4] | Ceramic resonator | CSTCC2M00G56-R0 | SMD | 2.0 | LV | (47) | (47) | 0 | 1.6 | 3.6 |
| | | CSTCR4M00G55-R0 | SMD | 4.0 | | (39) | (39) | 0 | | |
| | | CSTLS4M00G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCC2M00G56-R0 | SMD | 2.0 | LS | (47) | (47) | 0 | 1.8 | 3.6 |
| | | CSTCR4M00G55-R0 | SMD | 4.0 | | (39) | (39) | 0 | | |
| | | CSTLS4M00G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCR4M19G55-R0 | SMD | 4.194 | | (39) | (39) | 0 | | |
| | | CSTLS4M19G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCR4M91G53-R0 | SMD | 4.915 | | (15) | (15) | 0 | | |
| | | CSTLS4M91G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCR5M00G53-R0 | SMD | 5.0 | | (15) | (15) | 0 | | |
| | | CSTLS5M00G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCR6M00G53-R0 | SMD | 6.0 | | (15) | (15) | 0 | | |
| | | CSTLS6M00G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCE8M00G52-R0 | SMD | 8.0 | | (10) | (10) | 0 | | |
| | | CSTLS8M00G53-B0 | Lead | | | (15) | (15) | 0 | | |

**Notes 1.** The part numbers of the products that can operate at up to 105°C differ from those listed in the above table. For details, contact Murata Manufacturing Co., Ltd.

**2.** Set the flash operation mode by using CMODE1 and CMODE0 bits of the option byte (000C2H/010C2H).

**3.** Values in parentheses in the C1, C2 columns indicate an internal capacitance.

**4.** When using these oscillators, contact Murata Manufacturing Co., Ltd. (http://www.murata.com/index.html).

**Remark** Relationship between operation voltage width, operation frequency of CPU and operation mode is as below.

HS (High speed main) mode: 2.7 V ≤ $V_{DD}$ ≤ 3.6 V@1 MHz to 32 MHz (When X1 oscillation: 1 MHz to 20 MHz)
2.4 V ≤ $V_{DD}$ ≤ 3.6 V@1 MHz to 16 MHz

LS (Low speed main) mode: 1.8 V ≤ $V_{DD}$ ≤ 3.6 V@1 MHz to 8 MHz

LV (Low voltage main) mode: 1.6 V ≤ $V_{DD}$ ≤ 3.6 V@1 MHz to 4 MHz

**(1) X1 oscillation:** As of February, 2013 (2/3)

| Manufacturer | Resonator | Part Number[Note 1] | SMD/ Lead | Frequency (MHz) | Flash operation mode[Note 2] | Recommended Circuit Constants[Note 3] (reference) | | | Oscillation Voltage Range (V) | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | C1 (pF) | C2 (pF) | Rd (kΩ) | MIN. | MAX. |
| Murata Manufacturing Co., Ltd.[Note 4] | Ceramic resonator | CSTCC2M00G56-R0 | SMD | 2.0 | HS | (47) | (47) | 0 | 2.4 | 3.6 |
| | | CSTCR4M00G55-R0 | SMD | 4.0 | | (39) | (39) | 0 | | |
| | | CSTLS4M00G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCR4M19G55-R0 | SMD | 4.194 | | (39) | (39) | 0 | | |
| | | CSTLS4M19G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCR4M91G53-R0 | SMD | 4.915 | | (15) | (15) | 0 | | |
| | | CSTLS4M91G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCR5M00G53-R0 | SMD | 5.0 | | (15) | (15) | 0 | | |
| | | CSTLS5M00G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCR6M00G53-R0 | SMD | 6.0 | | (15) | (15) | 0 | | |
| | | CSTLS6M00G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCE8M00G52-R0 | SMD | 8.0 | | (10) | (10) | 0 | | |
| | | CSTLS8M00G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCE8M38G52-R0 | SMD | 8.388 | | (10) | (10) | 0 | | |
| | | CSTLS8M38G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCE10M0G52-R0 | SMD | 10.0 | | (10) | (10) | 0 | | |
| | | CSTLS10M0G53-B0 | Lead | | | (15) | (15) | 0 | | |
| | | CSTCE12M0G52-R0 | SMD | 12.0 | | (10) | (10) | 0 | | |
| | | CSTCE16M0V53-R0 | SMD | 16.0 | | (15) | (15) | 0 | | |
| | | CSTLS16M0X51-B0 | Lead | | | (5) | (5) | 0 | | |
| | | CSTCE20M0V51-R0 | SMD | 20.0 | | (5) | (5) | 0 | 2.7 | 3.6 |
| | | CSTLS20M0X51-B0 | Lead | | | (5) | (5) | 0 | | |

**Notes 1.** The part numbers of the products that can operate at up to 105°C differ from those listed in the above table. For details, contact Murata Manufacturing Co., Ltd.

**2.** Set the flash operation mode by using CMODE1 and CMODE0 bits of the option byte (000C2H/010C2H).

**3.** Values in parentheses in the C1, C2 columns indicate an internal capacitance.

**4.** When using these oscillators, contact Murata Manufacturing Co., Ltd. (http://www.murata.com/index.html).

**Remark** Relationship between operation voltage width, operation frequency of CPU and operation mode is as below.

HS (High speed main) mode: 2.7 V ≤ $V_{DD}$ ≤ 3.6 V@1 MHz to 32 MHz (When X1 oscillation: 1 MHz to 20 MHz)
2.4 V ≤ $V_{DD}$ ≤ 3.6 V@1 MHz to 16 MHz

LS (Low speed main) mode: 1.8 V ≤ $V_{DD}$ ≤ 3.6 V@1 MHz to 8 MHz

LV (Low voltage main) mode: 1.6 V ≤ $V_{DD}$ ≤ 3.6 V@1 MHz to 4 MHz

<R> **(1) X1 oscillation:** As of February, 2013 (3/3)

| Manufacturer | Resonator | Part Number | SMD/ Lead | Frequency (MHz) | Flash operation mode[Note 1] | Recommended Circuit Constants (reference) | | | Oscillation Voltage Range (V) | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | C1 (pF) | C2 (pF) | Rd (kΩ) | MIN. | MAX. |
| KYOCERA Corporation[Note 2] | Ceramic resonator | PBRV4.00MR50Y000 | SMD | 4.0 | LV | 15 | 15 | 0 | 1.6 | 3.6 |
| | | PBRV4.00MR50Y000 | SMD | 4.0 | LS | 15 | 15 | 0 | 1.8 | 3.6 |
| | | PRQV8.00CR5010Y000 | SMD | 8.0 | | 10 | 10 | 0 | | |
| | | PRQV8.00CR5010Y000 | SMD | 8.0 | HS | 10 | 10 | 0 | 2.4 | 3.6 |
| Nihon Dempa Kogyo Co., Ltd.[Note 3] | Crystal resonator | NX8045GE | SMD | 4.0 | LV | 6 | 6 | 0 | 1.6 | 3.6 |
| | | NX8045GE | SMD | 4.0 | LS | 6 | 6 | 0 | 1.8 | 3.6 |
| | | NX8045GB | SMD | 8.0 | | 0 | 0 | 0 | | |
| | | NX8045GB | SMD | 8.0 | HS | 0 | 0 | 0 | 2.4 | 3.6 |
| | | NX3225GB | SMD | 16.0 | | 1 | 1 | 0 | | |
| | | NX2520SA[Note 4] | SMD | 20.0 | | 0 | 0 | 0 | 2.7 | 3.6 |

**Notes 1.** Set the flash operation mode by using CMODE1 and CMODE0 bits of the option byte (000C2H/010C2H).

**2.** When using these oscillators, contact KYOCERA Corporation (http://global.kyocera.com/).

**3.** When using these oscillators, contact Nihon Dempa Kogyo Co., Ltd. (http://www.ndk.com/en/index.html).

**4.** This resonator can operate at up to 85°C. For details about the products that can operate at up to 105C, contact the manufacturer of the resonator.

**Remark** Relationship between operation voltage width, operation frequency of CPU and operation mode is as below.

HS (High speed main) mode: 2.7 V ≤ $V_{DD}$ ≤ 3.6 V@1 MHz to 32 MHz (When X1 oscillation: 1 MHz to 20 MHz)
2.4 V ≤ $V_{DD}$ ≤ 3.6 V@1 MHz to 16 MHz

LS (Low speed main) mode: 1.8 V ≤ $V_{DD}$ ≤ 3.6 V@1 MHz to 8 MHz

LV (Low voltage main) mode: 1.6 V ≤ $V_{DD}$ ≤ 3.6 V@1 MHz to 4 MHz

<R> **(2) XT1 oscillation:** As of July, 2013

| Manufacturer | Resonator | Part Number | SMD/ Lead | Frequency (MHz) | Load Capacitance CL (pF) | XT1 oscillation mode | Recommended Circuit Constants (reference) | | | Oscillation Voltage Range (V) | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | C3 (pF) | C4 (pF) | Rd (kΩ) | MIN. | MAX. |
| Nihon Dempa Kogyo Co., Ltd.[Note] | Crystal resonator | NX3215SA | SMD | 32.768 | 6.0 | Normal oscillation | 8 | 9 | 0 | 1.6 | 3.6 |
| | | | | | | Low power consumption oscillation | 8 | 8 | 0 | | |
| | | | | | | Ultra-low power consumption oscillation | 7 | 8 | 0 | | |
| | | NX2012SA | | 32.768 | 6.0 | Normal oscillation | 8 | 9 | 0 | | |
| | | | | | | Low power consumption oscillation | 8 | 8 | 0 | | |
| | | | | | | Ultra-low power consumption oscillation | 7 | 8 | 0 | | |

**Note** When using these oscillators, contact Nihon Dempa Kogyo Co., Ltd. (http://www.ndk.com/en/index.html).

# CHAPTER 6 TIMER ARRAY UNIT

The timer array unit is provided in all products (Unit 0, Channels 0 to 7).

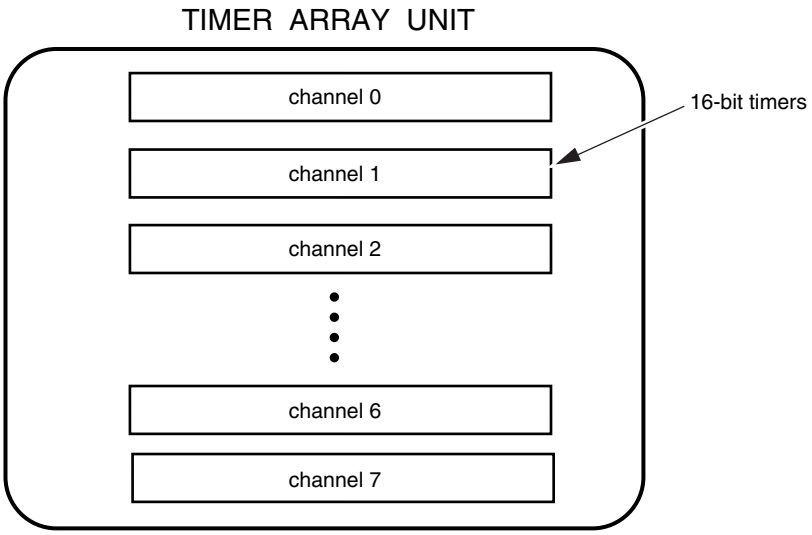
| Units | Channels | 25, 32, 48, 64-pin |
|---|---|---|
| Unit 0 | Channel 0 | √ |
| | Channel 1 | √ |
| | Channel 2 | √ |
| | Channel 3 | √ |
| | Channel 4 | √ |
| | Channel 5 | √ |
| | Channel 6 | √ |
| | Channel 7 | √ |

**Cautions 1. The presence or absence of timer I/O pins depends on the product. See Table 6-2 Timer I/O Pins provided in Each Product for details.**

**2. Most of the following descriptions in this chapter use the 64-pin products as an example.**

The timer array unit has eight 16-bit timers.

Each 16-bit timer is called a channel and can be used as an independent timer. In addition, two or more “channels” can be used to create a high-accuracy timer.




For details about each function, see the table below.

| Independent channel operation function | Simultaneous channel operation function |
|---|---|
| • Interval timer (→ see **6.8.1**)<br>• Square wave output (→ see **6.8.1**)<br>• External event counter (→ see **6.8.2**)<br>• Divider[Note] (→ see **6.8.3**)<br>• Input pulse interval measurement (→ see **6.8.4**)<br>• Measurement of high-/low-level width of input signal (→ see **6.8.5**)<br>• Delay counter (→ see **6.8.6**) | • One-shot pulse output(→ see **6.9.1**)<br>• PWM output(→ see **6.9.2**)<br>• Multiple PWM output(→ see **6.9.3**) |

**Note** Only channel 0 of unit 0.

It is possible to use the 16-bit timer of channels 1 and 3 of the units 0 as two 8-bit timers (higher and lower). The functions that can use channels 1 and 3 as 8-bit timers are as follows:

- Interval timer (higher/lower 8-bit timer only)/square wave output (lower 8-bit timer only)
- External event counter (lower 8-bit timer only)
- Delay counter (lower 8-bit timer only)

Channel 7 of unit 0 can be used to realize LIN-bus communication operating in combination with UART2 of the serial array unit (32, 48, and 64-pin products only).

## 6.1 Functions of Timer Array Unit

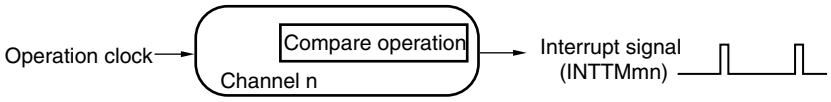
Timer array unit has the following functions.

### 6.1.1 Independent channel operation function

By operating a channel independently, it can be used for the following purposes without being affected by the operation mode of other channels.

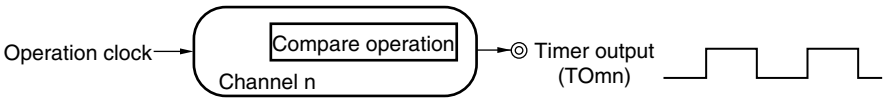
**(1) Interval timer**

Each timer of a unit can be used as a reference timer that generates an interrupt (INTTMmn) at fixed intervals.



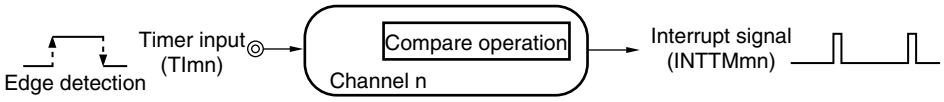

**(2) Square wave output**

A toggle operation is performed each time INTTMmn interrupt is generated and a square wave with a duty factor of 50% is output from a timer output pin (TOmn).



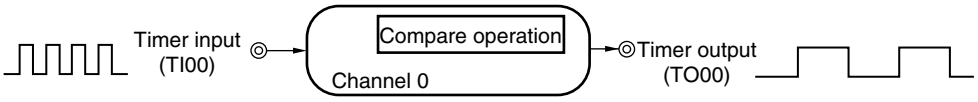

**(3) External event counter**

Each timer of a unit can be used as an event counter that generates an interrupt when the number of the valid edges of a signal input to the timer input pin (TImn) has reached a specific value.



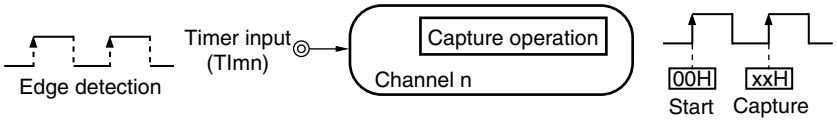

**(4) Divider function (channel 0 only)**

A clock input from a timer input pin (TI00) is divided and output from an output pin (TOm0).




**(5) Input pulse interval measurement**

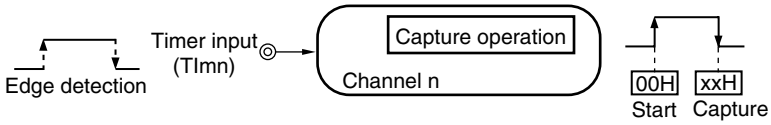
Counting is started by the valid edge of a pulse signal input to a timer input pin (TImn). The count value of the timer is captured at the valid edge of the next pulse. In this way, the interval of the input pulse can be measured.




(**Note**, **Caution,** and **Remark** are listed on the next page.)

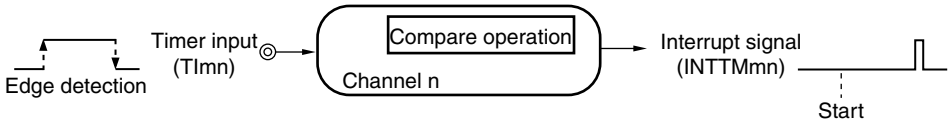
**(6) Measurement of high-/low-level width of input signal**

Counting is started by a single edge of the signal input to the timer input pin (TImn), and the count value is captured at the other edge. In this way, the high-level or low-level width of the input signal can be measured.




**(7) Delay counter**

Counting is started at the valid edge of the signal input to the timer input pin (TImn), and an interrupt is generated after any delay period.




**Remarks 1** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

**2.** The presence or absence of timer I/O pins of channel 0 to 7 depends on the product. See **Table 6-2 Timer I/O Pins provided in Each Product** for details.

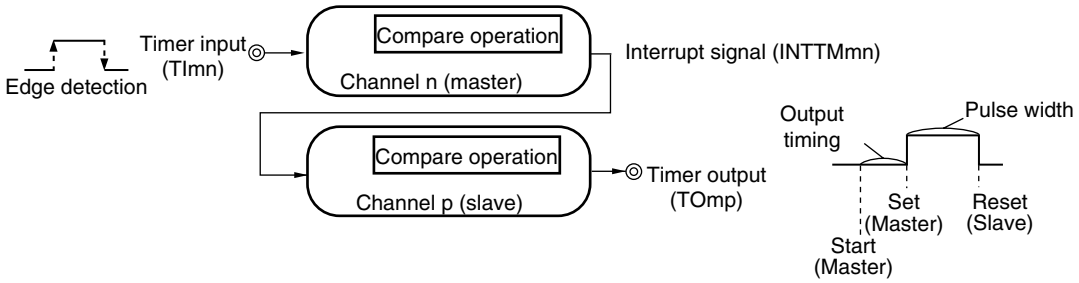
### 6.1.2 Simultaneous channel operation function

By using the combination of a master channel (a reference timer mainly controlling the cycle) and slave channels (timers operating according to the master channel), channels can be used for the following purposes.

**(1) One-shot pulse output**

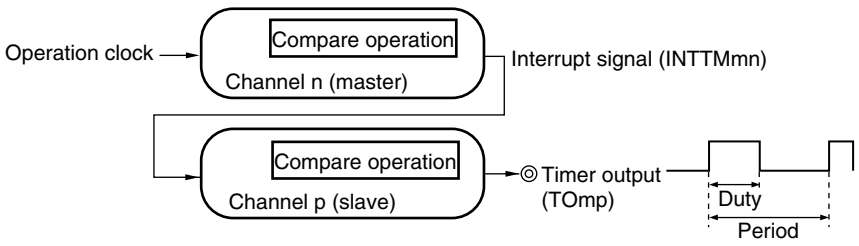
Two channels are used as a set to generate a one-shot pulse with a specified output timing and a specified pulse width.

<R>




**(2) PWM (Pulse Width Modulation) output**

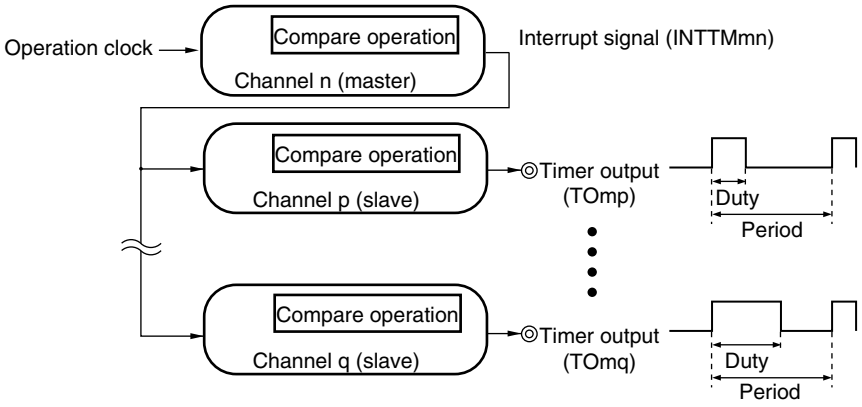
Two channels are used as a set to generate a pulse with a specified period and a specified duty factor.




(**Caution** is listed on the next page.)

**(3) Multiple PWM (Pulse Width Modulation) output**

By extending the PWM function and using one master channel and two or more slave channels, up to seven types of PWM signals that have a specific period and a specified duty factor can be generated.




**Caution For details about the rules of simultaneous channel operation function, see 6.4.1 Basic rules of simultaneous channel operation function.**

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7(however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7)), p, q: Slave channel number (n < p < q ≤ 7)

### 6.1.3 8-bit timer operation function (channels 1 and 3 only)

The 8-bit timer operation function makes it possible to use a 16-bit timer channel in a configuration consisting of two 8-bit timer channels. This function can only be used for channels 1 and 3.

**Caution There are several rules for using 8-bit timer operation function.**
**For details, see 6.4.2 Basic rules of 8-bit timer operation function (channels 1 and 3 only).**

### 6.1.4 LIN-bus supporting function (channel 7 of unit 0 only)

Timer array unit is used to check whether signals received in LIN-bus communication match the LIN-bus communication format.

**(1) Detection of wakeup signal**

The timer starts counting at the falling edge of a signal input to the serial data input pin (RxD2) of UART2 and the count value of the timer is captured at the rising edge. In this way, a low-level width can be measured. If the low-level width is greater than a specific value, it is recognized as a wakeup signal.

**(2) Detection of break field**

The timer starts counting at the falling edge of a signal input to the serial data input pin (RxD2) of UART2 after a wakeup signal is detected, and the count value of the timer is captured at the rising edge. In this way, a low-level width is measured. If the low-level width is greater than a specific value, it is recognized as a break field.

**(3) Measurement of pulse width of sync field**

After a break field is detected, the low-level width and high-level width of the signal input to the serial data input pin (RxD2) of UART2 are measured. From the bit interval of the sync field measured in this way, a baud rate is calculated.

**Remark** For details about setting up the operations used to implement the LIN-bus, see **6.3.13 Input switch control register (ISC)** and **6.8.5 Operation as input signal high-/low-level width measurement**.

## 6.2 Configuration of Timer Array Unit

Timer array unit includes the following hardware.

<R>

**Table 6-1. Configuration of Timer Array Unit**

| Item | Configuration |
|---|---|
| Timer/counter | Timer count register mn (TCRmn) |
| Register | Timer data register mn (TDRmn) |
| Timer input | TI00, TI01, TI03 to TI07[Note 1], RxD2 pin (for LIN-bus) |
| Timer output | TO00, TO01, TO03 to TO07[Note 1], output controller |
| Control registers | <Registers of unit setting block><br>• Peripheral enable register 0 (PER0)<br>• Timer clock select register m (TPSm)<br>• Timer channel enable status register m (TEm)<br>• Timer channel start register m (TSm)<br>• Timer channel stop register m (TTm)<br>• Timer input select register 0 (TIS0)<br>• Timer output enable register m (TOEm)<br>• Timer output register m (TOm)<br>• Timer output level register m (TOLm)<br>• Timer output mode register m (TOMm) |
| | <Registers of each channel><br>• Timer mode register mn (TMRmn)<br>• Timer status register mn (TSRmn)<br>• Input switch control register (ISC)<br>• Noise filter enable registers 1 (NFEN1)<br>• Port mode control register (PMCxx)[Note 2]<br>• Port mode register (PMxx)[Note 2]<br>• Port register (Pxx)[Note 2] |

**Notes 1.** The presence or absence of timer I/O pins of channel 0 to 7 depends on the product. See **Table 6-2 Timer I/O Pins provided in Each Product** for details.

**2.** The Port mode control register (PMCxx)., port mode registers (PMxx) and port registers (Pxx) to be set differ depending on the product. for details, see **4.5.3 Register setting examples for used port and alternate functions.**

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

The presence or absence of timer I/O pins in each timer array unit channel depends on the product.

**Table 6-2. Timer I/O Pins provided in Each Product**

| Timer array unit channels | | I/O Pins of Each Product | | | |
|---|---|---|---|---|---|
| | | 64-pin | 48-pin | 32-pin | 25-pin |
| Unit 0 | Channel 0 | TI00, TO00 | | | |
| | Channel 1 | TI01/TO01 | | – | |
| | Channel 2 | – | | | |
| | Channel 3 | TI03/TO03 | | | |
| | Channel 4 | TI04/TO04 | – | | |
| | Channel 5 | TI05/TO05 | – | | |
| | Channel 6 | TI06/TO06 | – | | |
| | Channel 7 | TI07/TO07 | | – | |

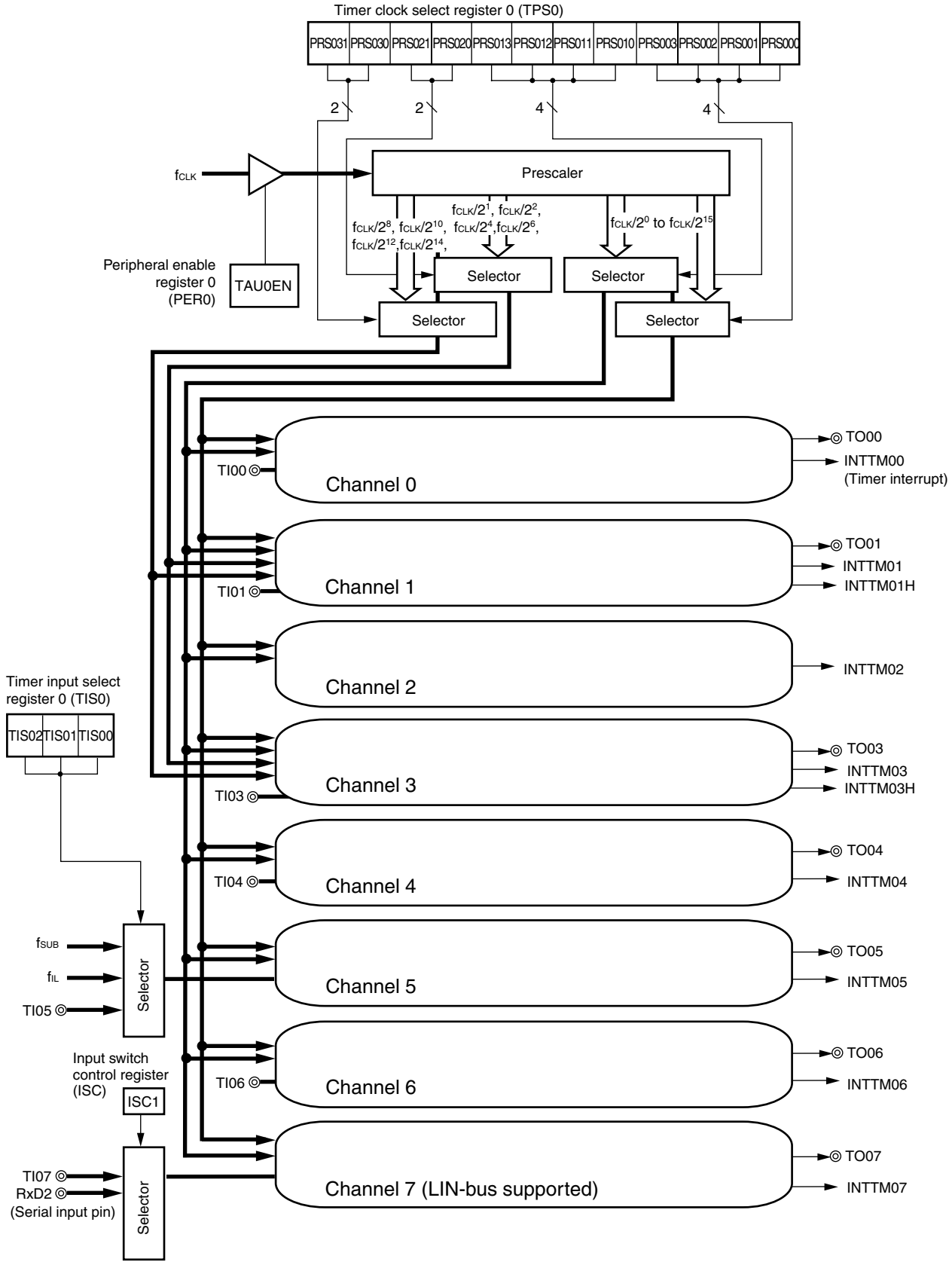
**Remarks 1.** When timer input and timer output are shared by the same pin, either only timer input or only timer output can be used.

**2.** –: There is no timer I/O pin, but the channel is available. (However, the channel can only be used as an interval timer.)

×: The channel is not available.

Figure 6-1 shows the block diagrams of the timer array unit.

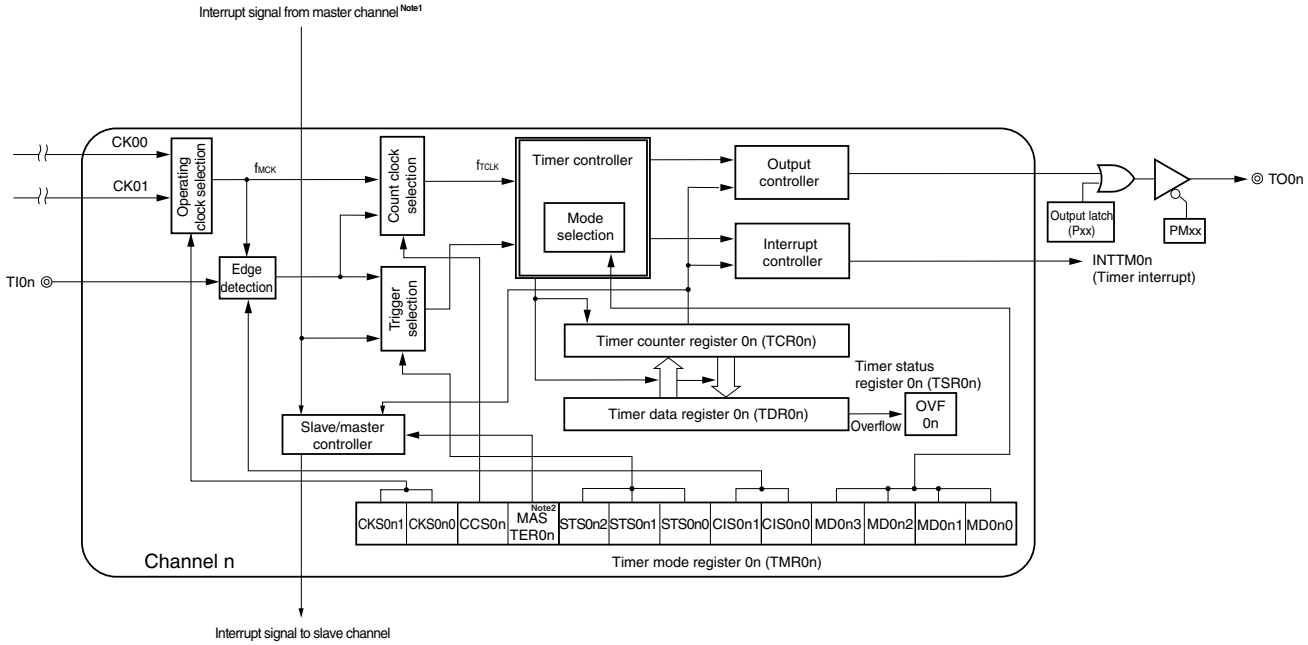
<R> **Figure 6-1. Entire Configuration of Timer Array Unit 0 (Example: 64-pin products)**




**Remark** $f_{SUB}$: Subsystem clock frequency

$f_{IL}$: Low-speed on-chip oscillator clock frequency

**Figure 6-2. Internal Block Diagram of Channel 0, 2, 4, 6 of Timer Array Unit 0**

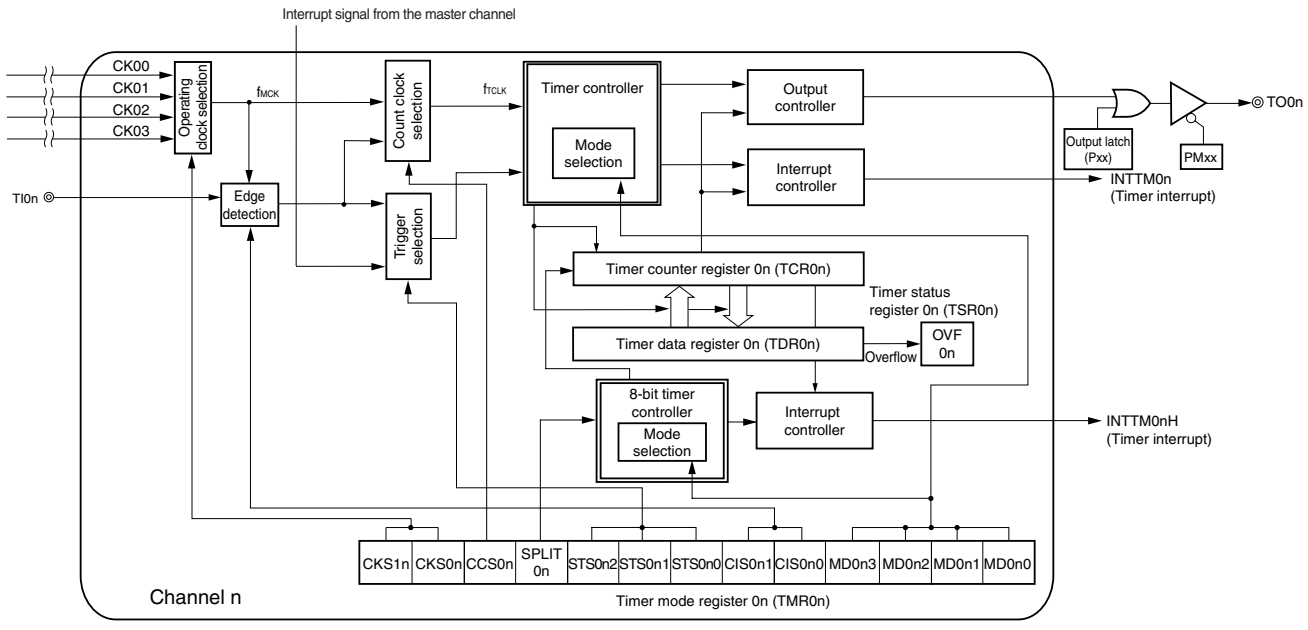



**Notes 1.** Channels 2, 4, 6 only
**2.** n = 2, 4, 6 only

**Remark** n = 0, 2, 4, 6

**Figure 6-3. Internal Block Diagram of Channels 1 and 3 of Timer Array Unit 0**

<R>




**Remark** n = 1, 3

**Figure 6-4. Internal Block Diagram of Channel 5 of Timer Array Unit 0**

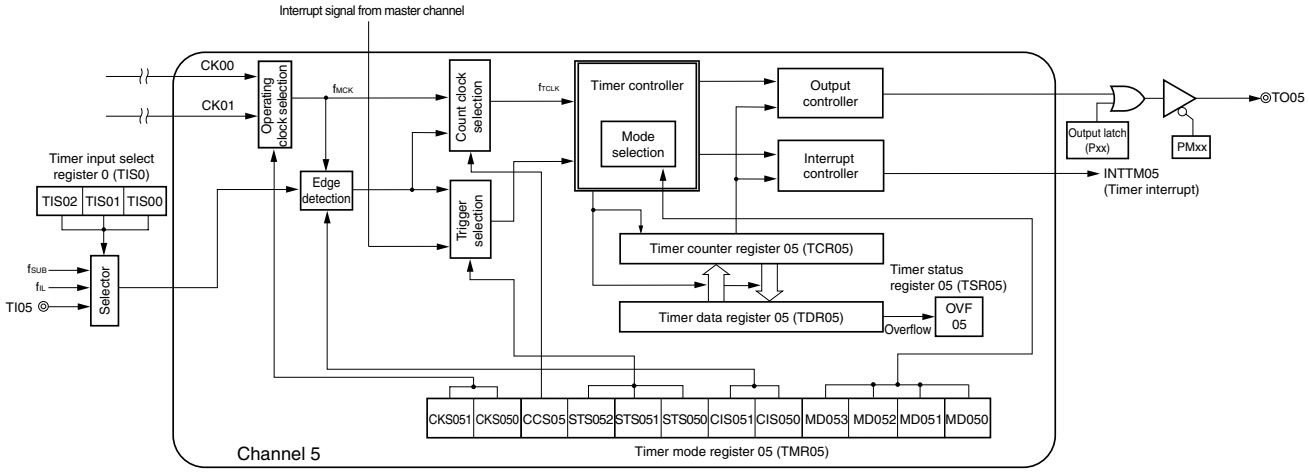
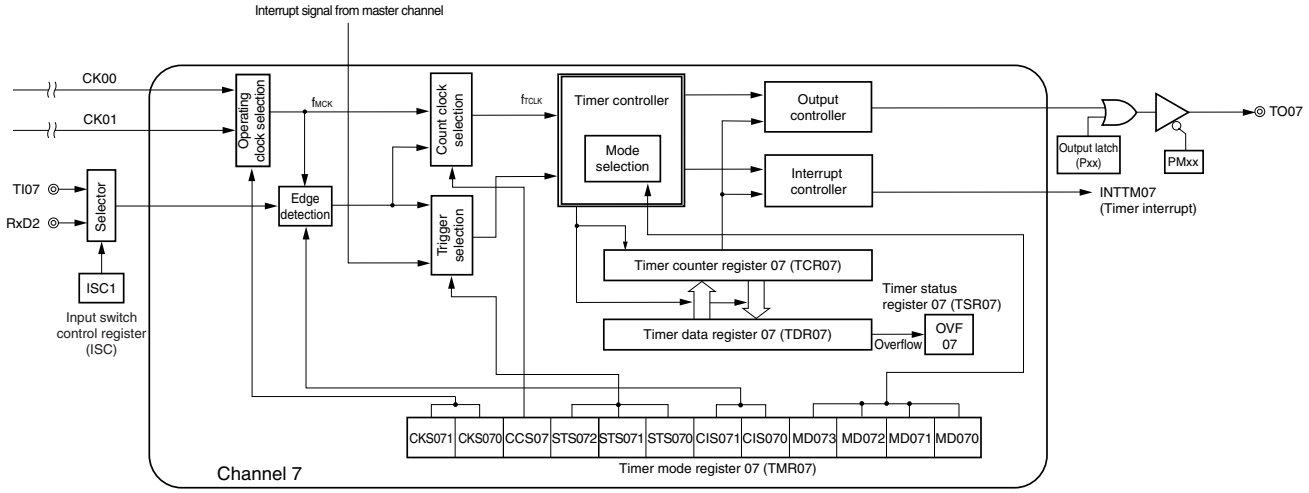



**Figure 6-5. Internal Block Diagram of Channel 7 of Timer Array Unit 0**



### 6.2.1 Timer count register mn (TCRmn)

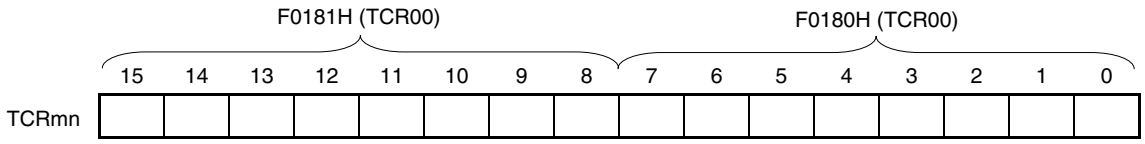
The TCRmn register is a 16-bit read-only register and is used to count clocks.

The value of this counter is incremented or decremented in synchronization with the rising edge of a count clock.

Whether the counter is incremented or decremented depends on the operation mode that is selected by the MDmn3 to MDmn0 bits of timer mode register mn (TMRmn) (see **6.3.3 Timer mode register mn (TMRmn)**).

**Figure 6-6. Format of Timer Count Register mn (TCRmn)**

Address: F0180H, F0181H (TCR00) to F018EH, F018FH (TCR07), F01C0H, F01C1H (TCR10) to F01CEH, F01CFH (TCR17) After reset: FFFFH R




**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

The count value can be read by reading timer count register mn (TCRmn).

The count value is set to FFFFH in the following cases.

- When the reset signal is generated
- When the TAUmEN bit of peripheral enable register 0 (PER0) is cleared
- When counting of the slave channel has been completed in the PWM output mode
- When counting of the slave channel has been completed in the delay count mode
- When counting of the master/slave channel has been completed in the one-shot pulse output mode
- When counting of the slave channel has been completed in the multiple PWM output mode

The count value is cleared to 0000H in the following cases.

- When the start trigger is input in the capture mode
- When capturing has been completed in the capture mode

**Caution The count value is not captured to timer data register mn (TDRmn) even when the TCRmn register is read.**

The TCRmn register read value differs as follows according to operation mode changes and the operating status.

**Table 6-3. Timer Count Register mn (TCRmn) Read Value in Various Operation Modes**

| Operation Mode | Count Mode | Timer count register mn (TCRmn) Read Value[Note] | | | |
|---|---|---|---|---|---|
| | | Value if the operation mode was changed after releasing reset | Value if the operation was count operation paused (TTmn = 1) | Value if the operation mode was changed after count operation paused (TTmn = 1) | Value when waiting for a start trigger after one count |
| Interval timer mode | Count down | FFFFH | Value if stop | Undefined | – |
| Capture mode | Count up | 0000H | Value if stop | Undefined | – |
| Event counter mode | Count down | FFFFH | Value if stop | Undefined | – |
| One-count mode | Count down | FFFFH | Value if stop | Undefined | FFFFH |
| Capture & one-count mode | Count up | 0000H | Value if stop | Undefined | Capture value of TDRmn register + 1 |

**Note** This indicates the value read from the TCRmn register when channel n has stopped operating as a timer (TEmn = 0) and has been enabled to operate as a counter (TSmn = 1). The read value is held in the TCRmn register until the count operation starts.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

### 6.2.2 Timer data register mn (TDRmn)

This is a 16-bit register from which a capture function and a compare function can be selected.

The capture or compare function can be switched by selecting an operation mode by using the MDmn3 to MDmn0 bits of timer mode register mn (TMRmn).

The value of the TDRmn register can be changed at any time.

This register can be read or written in 16-bit units.

In addition, for the TDRm1 and TDRm3 registers, while in the 8-bit timer mode (when the SPLIT bits of timer mode registers 01 and 03 (TMRm1, TMRm3) are 1), it is possible to read or write the data in 8-bit units, with TDRm1H and TDRm3H used as the higher 8 bits, and TDRm1L and TDRm3L used as the lower 8 bits.

Reset signal generation clears this register to 0000H.

**Figure 6-7. Format of Timer Data Register mn (TDRmn) (n = 0, 2, 4 to 7)**

Address: FFF18H, FFF19H (TDR00), FFF64H, FFF65H (TDR02), FFF68H, FFF69H (TDR04) to FFF6EH, FFF6FH (TDR07) After reset: 0000H R/W

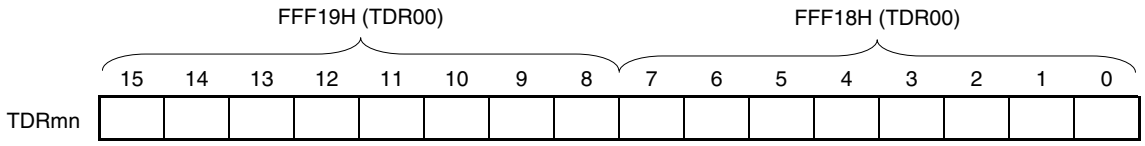
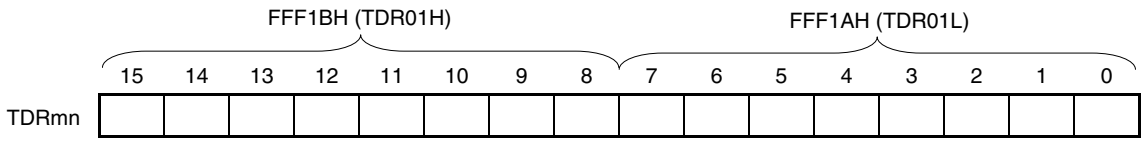



**Figure 6-8. Format of Timer Data Register mn (TDRmn) (n = 1, 3)**

Address: FFF1AH, FFF1BH (TDR01), FFF65H, FFF67H (TDR03) After reset: 0000H R/W




**(i) When timer data register mn (TDRmn) is used as compare register**

Counting down is started from the value set to the TDRmn register. When the count value reaches 0000H, an interrupt signal (INTTMmn) is generated. The TDRmn register holds its value until it is rewritten.

**Caution The TDRmn register does not perform a capture operation even if a capture trigger is input, when it is set to the compare function.**

**(ii) When timer data register mn (TDRmn) is used as capture register**

The count value of timer count register mn (TCRmn) is captured to the TDRmn register when the capture trigger is input.

A valid edge of the TImn pin can be selected as the capture trigger. This selection is made by timer mode register mn (TMRmn).

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

## 6.3 Registers Controlling Timer Array Unit

Timer array unit is controlled by the following registers.

- Peripheral enable register 0 (PER0)
- Timer clock select register m (TPSm)
- Timer mode register mn (TMRmn)
- Timer status register mn (TSRmn)
- Timer channel enable status register m (TEm)
- Timer channel start register m (TSm)
- Timer channel stop register m (TTm)
- Timer input select register 0 (TIS0)
- Timer output enable register m (TOEm)
- Timer output register m (TOm)
- Timer output level register m (TOLm)
- Timer output mode register m (TOMm)
- Input switch control register (ISC)
- Noise filter enable registers 1 (NFEN1)
- Port mode control register (PMCxx)
- Port mode register (PMxx)
- Port register (Pxx)

**Caution Which registers and bits are included depends on the product. Be sure to set bits that are not mounted to their initial values.**

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

### 6.3.1 Peripheral enable register 0 (PER0)

This registers is used to enable or disable supplying the clock to the peripheral hardware. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

When the timer array unit 0 is used, be sure to set bit 0 (TAU0EN) of this register to 1.

The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

<R> **Figure 6-9. Format of Peripheral Enable Register 0 (PER0)**

Address: F00F0H After reset: 00H R/W

| Symbol | <7> | 6 | <5> | <4> | <3> | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER0 | RTCEN | 0 | ADCEN | IICA0EN | SAU1EN[Note] | SAU0EN | 0 | TAU0EN |

| TAU0EN | Control of timer array 0 unit input clock |
|---|---|
| 0 | Stops supply of input clock.<br>• SFR used by the timer array unit 0 cannot be written.<br>• The timer array unit 0 is in the reset status. |
| 1 | Supplies input clock.<br>• SFR used by the timer array unit 0 can be read/written. |

**Notes** 32, 48, and 64-pin products only.

<R> **Cautions 1. When setting the timer array unit, be sure to set the TAUmEN bit to 1 first. If TAUmEN = 0, writing to a control register of timer array unit is ignored, and all read values are default values (except for the timer input select register 0 (TIS0), input switch control register (ISC), noise filter enable register 1 (NFEN1), port mode control registers 0, 1, 3, 4 (PMC0, PMC1, PMC3, PMC4), port mode registers 0, 1, 3, 4 (PM0, PM1, PM3, PM4), and port registers 0, 1, 3, 4 (P0, P1, P3, P4)).**

- **Timer clock select register m (TPSm)**
- **Timer mode register mn (TMRmn)**
- **Timer status register mn (TSRmn)**
- **Timer channel enable status register m (TEm)**
- **Timer channel start register m (TSm)**
- **Timer channel stop register m (TTm)**
- **Timer output enable register m (TOEm)**
- **Timer output register m (TOm)**
- **Timer output level register m (TOLm)**
- **Timer output mode register m (TOMm)**

**2. Be sure to clear the following bits to 0.**
**25-pin products: bits 1, 3, 6**
**32, 48, 64-pin products: bits 1, 6**

### 6.3.2 Timer clock select register m (TPSm)

<R> The TPSm register is a 16-bit register that is used to select two types or four types of operation clocks (CKm0, CKm1, CKm2, CKm3) that are commonly supplied to each channel. CKm0 is selected by using bits 3 to 0 of the TPSm register, and CKm1 is selected by using bits 7 to 4 of the TPSm register. In addition, only for channel 1 and 3, CKm2 and CKm3 can be also selected. CKm2 is selected by using bits 9 and 8 of the TPSm register, and CKm3 is selected by using bits 13 and 12 of the TPSm register.

Rewriting of the TPSm register during timer operation is possible only in the following cases.

If the PRSm00 to PRSm03 bits can be rewritten (n = 0 to 7):

All channels for which CKm0 is selected as the operation clock (CKSmn1, CKSmn0 = 0, 0) are stopped (TEmn = 0).

If the PRSm10 to PRSm13 bits can be rewritten (n = 0 to 7):

All channels for which CKm1 is selected as the operation clock (CKSmn1, CKSmn0 = 0, 1) are stopped (TEmn = 0).

If the PRSm20 and PRSm21 bits can be rewritten (n = 1, 3):

All channels for which CKm2 is selected as the operation clock (CKSmn1, CKSmn0 = 1, 0) are stopped (TEmn = 0).

If the PRSm30 and PRSm31 bits can be rewritten (n = 1, 3):

All channels for which CKm3 is selected as the operation clock (CKSmn1, CKSmn0 = 1, 1) are stopped (TEmn = 0).

The TPSm register can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0000H.

**Figure 6-10. Format of Timer Clock Select register m (TPSm) (1/2)**

Address: F01B6H, F01B7H (TPS0) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TPSm | 0 | 0 | PRS m31 | PRS m30 | 0 | 0 | PRS m21 | PRS m20 | PRS m13 | PRS m12 | PRS m11 | PRS m10 | PRS m03 | PRS m02 | PRS m01 | PRS m00 |

| PRS mk3 | PRS mk2 | PRS mk1 | PRS mk0 | Selection of operation clock (CKmk)[Note] (k = 0, 1) | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | $f_{CLK}$ = 2 MHz | $f_{CLK}$ = 5 MHz | $f_{CLK}$ = 10 MHz | $f_{CLK}$ = 20 MHz | $f_{CLK}$ = 32 MHz |
| 0 | 0 | 0 | 0 | $f_{CLK}$ | 2 MHz | 5 MHz | 10 MHz | 20 MHz | 32 MHz |
| 0 | 0 | 0 | 1 | $f_{CLK}/2$ | 1 MHz | 2.5 MHz | 5 MHz | 10 MHz | 16 MHz |
| 0 | 0 | 1 | 0 | $f_{CLK}/2^2$ | 500 kHz | 1.25 MHz | 2.5 MHz | 5 MHz | 8 MHz |
| 0 | 0 | 1 | 1 | $f_{CLK}/2^3$ | 250 kHz | 625 kHz | 1.25 MHz | 2.5 MHz | 4 MHz |
| 0 | 1 | 0 | 0 | $f_{CLK}/2^4$ | 125 kHz | 313 kHz | 625 kHz | 1.25 MHz | 2 MHz |
| 0 | 1 | 0 | 1 | $f_{CLK}/2^5$ | 62.5 kHz | 156 kHz | 313 kHz | 625 kHz | 1 MHz |
| 0 | 1 | 1 | 0 | $f_{CLK}/2^6$ | 31.3 kHz | 78.1 kHz | 156 kHz | 313 kHz | 500 kHz |
| 0 | 1 | 1 | 1 | $f_{CLK}/2^7$ | 15.6 kHz | 39.1 kHz | 78.1 kHz | 156 kHz | 250 kHz |
| 1 | 0 | 0 | 0 | $f_{CLK}/2^8$ | 7.81 kHz | 19.5 kHz | 39.1 kHz | 78.1 kHz | 125 kHz |
| 1 | 0 | 0 | 1 | $f_{CLK}/2^9$ | 3.91 kHz | 9.77 kHz | 19.5 kHz | 39.1 kHz | 62.5 kHz |
| 1 | 0 | 1 | 0 | $f_{CLK}/2^{10}$ | 1.95 kHz | 4.88 kHz | 9.77 kHz | 19.5 kHz | 31.3 kHz |
| 1 | 0 | 1 | 1 | $f_{CLK}/2^{11}$ | 977 Hz | 2.44 kHz | 4.88 kHz | 9.77 kHz | 15.6 kHz |
| 1 | 1 | 0 | 0 | $f_{CLK}/2^{12}$ | 488 Hz | 1.22 kHz | 2.44 kHz | 4.88 kHz | 7.81 kHz |
| 1 | 1 | 0 | 1 | $f_{CLK}/2^{13}$ | 244 Hz | 610 Hz | 1.22 kHz | 2.44 kHz | 3.91 kHz |
| 1 | 1 | 1 | 0 | $f_{CLK}/2^{14}$ | 122 Hz | 305 Hz | 610 Hz | 1.22 kHz | 1.95 kHz |
| 1 | 1 | 1 | 1 | $f_{CLK}/2^{15}$ | 61.0 Hz | 153 Hz | 305 Hz | 610 Hz | 977 Hz |

**Note** When changing the clock selected for $f_{CLK}$ (by changing the system clock control register (CKC) value), stop timer array unit (TTm = 00FFH).

**Cautions 1. Be sure to clear bits 15, 14, 11, 10 to "0".**

<R> **2. If $f_{CLK}$ (undivided) is selected as the operation clock (CKmk) and TDRnm is set to 0000H (n = 0 or 1, m = 0 to 7), interrupt requests output from timer array units cannot be used.**

**Remarks 1.** $f_{CLK}$: CPU/peripheral hardware clock frequency

**2.** Waveform of the clock to be selected in the TPSm register which becomes high level for one period of $f_{CLK}$ from its rising edge (m = 1 to 15). For details, see **6.5.1 Count clock ($f_{TCLK}$).**

**Figure 6-10. Format of Timer Clock Select register m (TPSm) (2/2)**

Address: F01B6H, F01B7H (TPS0) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TPSm | 0 | 0 | PRS m31 | PRS m30 | 0 | 0 | PRS m21 | PRS m20 | PRS m13 | PRS m12 | PRS m11 | PRS m10 | PRS m03 | PRS m02 | PRS m01 | PRS m00 |

| PRS m21 | PRS m20 | Selection of operation clock (CKm2) ote | | | | | |
|---|---|---|---|---|---|---|---|
| | | | $f_{CLK}$ = 2 MHz | $f_{CLK}$ = 5 MHz | $f_{CLK}$ = 10 MHz | $f_{CLK}$ = 20 MHz | $f_{CLK}$ = 32 MHz |
| 0 | 0 | $f_{CLK}/2$ | 1 MHz | 2.5 MHz | 5 MHz | 10 MHz | 16 MHz |
| 0 | 1 | $f_{CLK}/2^2$ | 500 kHz | 1.25 MHz | 2.5 MHz | 5 MHz | 8 MHz |
| 1 | 0 | $f_{CLK}/2^4$ | 125 kHz | 313 kHz | 625 MHz | 1.25 MHz | 2 MHz |
| 1 | 1 | $f_{CLK}/2^6$ | 31.3 kHz | 78.1 kHz | 156 kHz | 313 kHz | 500 kHz |

| PRS m31 | PRS m30 | Selection of operation clock (CKm3) Note | | | | | |
|---|---|---|---|---|---|---|---|
| | | | $f_{CLK}$ = 2 MHz | $f_{CLK}$ = 5 MHz | $f_{CLK}$ = 10 MHz | $f_{CLK}$ = 20 MHz | $f_{CLK}$ = 32 MHz |
| 0 | 0 | $f_{CLK}/2^8$ | 7.81 kHz | 19.5 kHz | 39.1 kHz | 78.1 kHz | 125 kHz |
| 0 | 1 | $f_{CLK}/2^{10}$ | 1.95 kHz | 4.88 kHz | 9.77 kHz | 19.5 kHz | 31.3 kHz |
| 1 | 0 | $f_{CLK}/2^{12}$ | 488 Hz | 1.22 kHz | 2.44 kHz | 4.88 kHz | 7.81 kHz |
| 1 | 1 | $f_{CLK}/2^{14}$ | 122 Hz | 305 Hz | 610 Hz | 1.22 kHz | 1.95 kHz |

**Note** When changing the clock selected for $f_{CLK}$ (by changing the system clock control register (CKC) value), stop timer array unit (TTm = 00FFH).
The timer array unit must also be stopped if the operating clock ($f_{MCK}$) or the valid edge of the signal input from the TImn pin is selected.

**Caution Be sure to clear bits 15, 14, 11, 10 to "0".**

By using channels 1 and 3 in the 8-bit timer mode and specifying CKm2 or CKm3 as the operation clock, the interval times shown in Table 6-4 can be achieved by using the interval timer function.

**Table 6-4. Interval Times Available for Operation Clock CKSm2 or CKSm3**

| Clock | | Interval time Note ($f_{CLK}$ = 32 MHz) | | | |
|---|---|---|---|---|---|
| | | 10 $\mu$s | 100 $\mu$s | 1 ms | 10 ms |
| CKm2 | $f_{CLK}/2$ | √ | – | – | – |
| | $f_{CLK}/2^2$ | √ | – | – | – |
| | $f_{CLK}/2^4$ | √ | √ | – | – |
| | $f_{CLK}/2^6$ | √ | √ | – | – |
| CKm3 | $f_{CLK}/2^8$ | – | √ | √ | – |
| | $f_{CLK}/2^{10}$ | – | √ | √ | – |
| | $f_{CLK}/2^{12}$ | – | – | √ | √ |
| | $f_{CLK}/2^{14}$ | – | – | √ | √ |

**Note** The margin is within 5 %.

**Remarks 1.** $f_{CLK}$: CPU/peripheral hardware clock frequency

**2.** For details of asignal of $f_{CLK}/2^j$ selected with the TPSm register, see **6.5.1 Count clock ($f_{TCLK}$).**

### 6.3.3 Timer mode register mn (TMRmn)

The TMRmn register sets an operation mode of channel n. This register is used to select the operation clock ($f_{MCK}$), select the count clock, select the master/slave, select the 16 or 8-bit timer (only for channels 1 and 3), specify the start trigger and capture trigger, select the valid edge of the timer input, and specify the operation mode (interval, capture, event counter, one-count, or capture and one-count).

Rewriting the TMRmn register is prohibited when the register is in operation (when TEmn = 1). However, bits 7 and 6 (CISmn1, CISmn0) can be rewritten even while the register is operating with some functions (when TEmn = 1) (for details, see **6.8 Independent Channel Operation Function of Timer Array Unit** and **6.9 Simultaneous Channel Operation Function of Timer Array Unit**.

The TMRmn register can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0000H.

**Caution The bits mounted depend on the channels in the bit 11 of TMRmn register.**

| | |
|---|---|
| **TMRm2, TMRm4, TMRm6:** | **MASTERmn bit (n = 2, 4, 6)** |
| **TMRm1, TMRm3:** | **SPLITmn bit (n = 1, 3)** |
| **TMRm0, TMRm5, TMRm7:** | **Fixed to 0** |

**Figure 6-11. Format of Timer Mode Register mn (TMRmn) (1/4)**

Address: F0190H, F0191H (TMR00) to F019EH, F019FH (TMR07) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 2, 4, 6 ) | CKS mn1 | CKS mn0 | 0 | CCS mn | MAST ERmn | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 1, 3) | CKS mn1 | CKS mn0 | 0 | CCS mn | SPLIT mn | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 0, 5, 7) | CKS mn1 | CKS mn0 | 0 | CCS mn | 0[Note] | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| CKS mn1 | CKS mn0 | Selection of operation clock ($f_{MCK}$) of channel n |
|---|---|---|
| 0 | 0 | Operation clock CKm0 set by timer clock select register m (TPSm) |
| 0 | 1 | Operation clock CKm2 set by timer clock select register m (TPSm) |
| 1 | 0 | Operation clock CKm1 set by timer clock select register m (TPSm) |
| 1 | 1 | Operation clock CKm3 set by timer clock select register m (TPSm) |
| Operation clock ($f_{MCK}$ ) is used by the edge detector. A count clock ($f_{TCLK}$) and a sampling clock are generated depending on the setting of the CCSmn bit.<br>The operation clocks CKm2 and CKm3 can only be selected for channels 1 and 3. | | |

| CCS mn | Selection of count clock ($f_{TCLK}$) of channel n |
|---|---|
| 0 | Operation clock ($f_{MCK}$) specified by the CKSmn0 and CKSmn1 bits |
| 1 | Valid edge of input signal input from the TImn pin<br>In channel 5, Valid edge of input signal selected by TIS0 |
| Count clock ($f_{TCLK}$) is used for the timer/counter, output controller, and interrupt controller. | |

**Note** Bit 11 is fixed at 0 of read only, write is ignored.

**Cautions 1. Be sure to clear bits 13, 5, and 4 to "0".**

**2. The timer array unit must be stopped (TTm = 00FFH) if the clock selected for $f_{CLK}$ is changed (by changing the value of the system clock control register (CKC)), even if the operating clock specified by using the CKSmn0 and CKSmn1 bits ($f_{MCK}$) or the valid edge of the signal input from the TImn pin is selected as the count clock ($f_{TCLK}$).**

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

**Figure 6-11. Format of Timer Mode Register mn (TMRmn) (2/4)**

Address: F0190H, F0191H (TMR00) to F019EH, F019FH (TMR07) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 2, 4, 6 ) | CKS mn1 | CKS mn0 | 0 | CCS mn | MASTERmn | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 1, 3) | CKS mn1 | CKS mn0 | 0 | CCS mn | SPLIT mn | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 0, 5, 7) | CKS mn1 | CKS mn0 | 0 | CCS mn | 0[Note] | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

(Bit 11 of TMRmn (n = 2, 4, 6))

| MASTER mn | Selection between using channel n independently or simultaneously with another channel(as a slave or master) |
|---|---|
| 0 | Operates in independent channel operation function or as slave channel in simultaneous channel operation function. |
| 1 | Operates as master channel in simultaneous channel operation function. |
| Only the channel 2, 4, 6 can be set as a master channel (MASTERmn = 1).<br>Be sure to use channel 0, 5, 7 are fixed to 0 (Regardless of the bit setting, channel 0 operates as master, because it is the highest channel).<br>Clear the MASTERmn bit to 0 for a channel that is used with the independent channel operation function. | |

(Bit 11 of TMRmn (n = 1, 3))

| SPLITmn | Selection of 8 or 16-bit timer operation for channels 1 and 3 |
|---|---|
| 0 | Operates as 16-bit timer.<br>(Operates in independent channel operation function or as slave channel in simultaneous channel operation function.) |
| 1 | Operates as 8-bit timer. |

| STS mn2 | STS mn1 | STS mn0 | Setting of start trigger or capture trigger of channel n |
|---|---|---|---|
| 0 | 0 | 0 | Only software trigger start is valid (other trigger sources are unselected). |
| 0 | 0 | 1 | Valid edge of the TImn pin input is used as both the start trigger and capture trigger. |
| 0 | 1 | 0 | Both the edges of the TImn pin input are used as a start trigger and a capture trigger. |
| 1 | 0 | 0 | Interrupt signal of the master channel is used (when the channel is used as a slave channel with the simultaneous channel operation function). |
| Other than above | | | Setting prohibited |

**Note** Bit 11 is fixed at 0 of read only, write is ignored.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

**Figure 6-11. Format of Timer Mode Register mn (TMRmn) (3/4)**

Address: F0190H, F0191H (TMR00) to F019EH, F019FH (TMR07) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 2, 4, 6 ) | CKS mn1 | CKS mn0 | 0 | CCS mn | MAST ERmn | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 1, 3) | CKS mn1 | CKS mn0 | 0 | CCS mn | SPLIT mn | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 0, 5, 7) | CKS mn1 | CKS mn0 | 0 | CCS mn | 0[Note] | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| CIS mn1 | CIS mn0 | Selection of TImn pin input valid edge |
|---|---|---|
| 0 | 0 | Falling edge |
| 0 | 1 | Rising edge |
| 1 | 0 | Both edges (when low-level width is measured)<br>Start trigger: Falling edge, Capture trigger: Rising edge |
| 1 | 1 | Both edges (when high-level width is measured)<br>Start trigger: Rising edge, Capture trigger: Falling edge |
| If both the edges are specified when the value of the STSmn2 to STSmn0 bits is other than 010B, set the CISmn1 to CISmn0 bits to 10B. | | |

| MD mn3 | MD mn2 | MD mn1 | Operation mode of channel n | Corresponding function | Count operation of TCR |
|---|---|---|---|---|---|
| 0 | 0 | 0 | Interval timer mode | Interval timer / Square wave output / Divider function / PWM output (master) | Counting down |
| 0 | 1 | 0 | Capture mode | Input pulse interval measurement | Counting up |
| 0 | 1 | 1 | Event counter mode | External event counter | Counting down |
| 1 | 0 | 0 | One-count mode | Delay counter / One-shot pulse output / PWM output (slave) | Counting down |
| 1 | 1 | 0 | Capture & one-count mode | Measurement of high-/low-level width of input signal | Counting up |
| Other than above | | | Setting prohibited | | |
| The operation of each mode varies depending on MDmn0 bit (see next table). | | | | | |

**Note** Bit 11 is fixed at 0 of read only, write is ignored.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

**Figure 6-11. Format of Timer Mode Register mn (TMRmn) (4/4)**

Address: F0190H, F0191H (TMR00) to F019EH, F019FH (TMR07) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 2, 4, 6) | CKS mn1 | CKS mn0 | 0 | CCS mn | MAST ERmn | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 1, 3) | CKS mn1 | CKS mn0 | 0 | CCS mn | SPLIT mn | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn (n = 0, 5, 7) | CKS mn1 | CKS mn0 | 0 | CCS mn | 0[Note] | STS mn2 | STS mn1 | STS mn0 | CIS mn1 | CIS mn0 | 0 | 0 | MD mn3 | MD mn2 | MD mn1 | MD mn0 |

| Operation mode (Value set by the MDmn3 to MDmn1 bits (see table shown in the previous page)) | MD mn0 | Setting of starting counting and interrupt |
|---|---|---|
| • Interval timer mode (0, 0, 0) | 0 | Timer interrupt is not generated when counting is started (timer output does not change, either). |
| • Capture mode (0, 1, 0) | 1 | Timer interrupt is generated when counting is started (timer output also changes). |
| • Event counter mode (0, 1, 1) | 0 | Timer interrupt is not generated when counting is started (timer output does not change, either). |
| • One-count mode[Note 1] (1, 0, 0) | 0 | Start trigger is invalid during counting operation. At that time, interrupt is not generated. |
| | 1 | Start trigger is valid during counting operation[Note 2]. At that time, interrupt is not generated. |
| • Capture & one-count mode (1, 1, 0) | 0 | Timer interrupt is not generated when counting is started (timer output does not change, either). Start trigger is invalid during counting operation. At that time interrupt is not generated. |
| Other than above | | Setting prohibited |

**Notes 1.** Bit 11 is fixed at 0 of read only, write is ignored.

**2.** In one-count mode, interrupt output (INTTMmn) when starting a count operation and TOmn output are not controlled.

**3.** If the start trigger (TSmn = 1) is issued during operation, the counter is initialized, and recounting is started (does not occur the interrupt request).

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

### 6.3.4 Timer status register mn (TSRmn)

The TSRmn register indicates the overflow status of the counter of channel n.

The TSRmn register is valid only in the capture mode (MDmn3 to MDmn1 = 010B) and capture & one-count mode (MDmn3 to MDmn1 = 110B). See Table 6-5 for the operation of the OVF bit in each operation mode and set/clear conditions.

The TSRmn register can be read by a 16-bit memory manipulation instruction.

The lower 8 bits of the TSRmn register can be set with an 8-bit memory manipulation instruction with TSRmnL.

Reset signal generation clears this register to 0000H.

**Figure 6-12. Format of Timer Status Register mn (TSRmn)**

Address: F01A0H, F01A1H (TSR00) to F01AEH, F01AFH (TSR07) After reset: 0000H R

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TSRmn | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | OVF |

| OVF | Counter overflow status of channel n |
|---|---|
| 0 | Overflow does not occur. |
| 1 | Overflow occurs. |
| When OVF = 1, this flag is cleared (OVF = 0) when the next value is captured without overflow. | |

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

**Table 6-5. OVF Bit Operation and Set/Clear Conditions in Each Operation Mode**

| Timer operation mode | OVF bit | Set/clear conditions |
|---|---|---|
| • Capture mode | clear | When no overflow has occurred upon capturing |
| • Capture & one-count mode | set | When an overflow has occurred upon capturing |
| • Interval timer mode<br>• Event counter mode<br>• One-count mode | clear | – |
| | set | (Use prohibited) |

**Remark** The OVF bit does not change immediately after the counter has overflowed, but changes upon the subsequent capture.

### 6.3.5 Timer channel enable status register m (TEm)

The TEm register is used to enable or stop the timer operation of each channel.

Each bit of the TEm register corresponds to each bit of the timer channel start register m (TSm) and the timer channel stop register m (TTm). When a bit of the TSm register is set to 1, the corresponding bit of this register is set to 1. When a bit of the TTm register is set to 1, the corresponding bit of this register is cleared to 0.

The TEm register can be read by a 16-bit memory manipulation instruction.

The lower 8 bits of the TEm register can be set with a 1-bit or 8-bit memory manipulation instruction with TEmL.

Reset signal generation clears this register to 0000H.

**Figure 6-13. Format of Timer Channel Enable Status register m (TEm)**

Address: F01B0H, F01B1H (TE0) After reset: 0000H R

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TEm | 0 | 0 | 0 | 0 | TEHm3 | 0 | TEHm1 | 0 | TEm7 | TEm6 | TEm5 | TEm4 | TEm3 | TEm2 | TEm1 | TEm0 |

| TEH03 | Indication of whether operation of the higher 8-bit timer is enabled or stopped when channel 3 is in the 8-bit timer mode |
|---|---|
| 0 | Operation is stopped. |
| 1 | Operation is enabled. |

| TEH01 | Indication of whether operation of the higher 8-bit timer is enabled or stopped when channel 1 is in the 8-bit timer mode |
|---|---|
| 0 | Operation is stopped. |
| 1 | Operation is enabled. |

| TEmn | Indication of operation enable/stop status of channel n |
|---|---|
| 0 | Operation is stopped. |
| 1 | Operation is enabled. |
| This bit displays whether operation of the lower 8-bit timer for TEm1 and TEm3 is enabled or stopped when channel 1 or 3 is in the 8-bit timer mode. | |

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7)

### 6.3.6 Timer channel start register m (TSm)

The TSm register is a trigger register that is used to initialize timer count register mn (TCRmn) and start the counting operation of each channel.

When a bit of this register is set to 1, the corresponding bit of timer channel enable status register m (TEm) is set to 1. The TSmn, TSHm1, TSHm3 bits are immediately cleared when operation is enabled (TEmn, TEHm1, TEHm3 = 1), because they are trigger bits.

The TSm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the TSm register can be set with a 1-bit or 8-bit memory manipulation instruction with TSmL.

Reset signal generation clears this register to 0000H.

**Figure 6-14. Format of Timer Channel Start register m (TSm)**

Address: F01B2H, F01B3H (TS0) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TSm | 0 | 0 | 0 | 0 | TSHm3 | 0 | TSHm1 | 0 | TSm7 | TSm6 | TSm5 | TSm4 | TSm3 | TSm2 | TSm1 | TSm0 |

| TSHm3 | Trigger to enable operation (start operation) of the higher 8-bit timer when channel 3 is in the 8-bit timer mode |
|---|---|
| 0 | No trigger operation |
| 1 | The TEHm3 bit is set to 1 and the count operation becomes enabled.<br>The TCRm3 register count operation start in the interval timer mode in the count operation enabled state (see Table 6-6 in **6.5.2 Start timing of counter**). |

| TSHm1 | Trigger to enable operation (start operation) of the higher 8-bit timer when channel 1 is in the 8-bit timer mode |
|---|---|
| 0 | No trigger operation |
| 1 | The TEHm1 bit is set to 1 and the count operation becomes enabled.<br>The TCRm1 register count operation start in the interval timer mode in the count operation enabled state (see Table 6-6 in **6.5.2 Start timing of counter**). |

| TSmn | Operation enable (start) trigger of channel n |
|---|---|
| 0 | No trigger operation |
| 1 | The TEmn bit is set to 1 and the count operation becomes enabled.<br>The TCRmn register count operation start in the count operation enabled state varies depending on each operation mode (see Table 6-6 in **6.5.2 Start timing of counter**).<br>This bit is the trigger to enable operation (start operation) of the lower 8-bit timer for TSm1 and TSm3 when channel 1 or 3 is in the 8-bit timer mode. |

**Cautions 1. Be sure to clear bits 15 to 12, 10, 8 to "0"**

**2. When switching from a function that does not use TImn pin input to one that does, the following wait period is required from when timer mode register mn (TMRmn) is set until the TSmn (TSHm1, TSHm3) bit is set to 1.**

**When the TImn pin noise filter is enabled (TNFENmn = 1): Four cycles of the operation clock ($f_{MCK}$)**

**When the TImn pin noise filter is disabled (TNFENmn = 0): Two cycles of the operation clock ($f_{MCK}$)**

**Remarks 1.** When the TSm register is read, 0 is always read.

**2.** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

### 6.3.7 Timer channel stop register m (TTm)

The TTm register is a trigger register that is used to stop the counting operation of each channel.

When a bit of this register is set to 1, the corresponding bit of timer channel enable status register m (TEm) is cleared to 0. The TTmn, TTHm1, TTHm3 bits are immediately cleared when operation is stopped (TEmn, TEHm1, TEHm3 = 0), because they are trigger bits.

The TTm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the TTm register can be set with a 1-bit or 8-bit memory manipulation instruction with TTmL.

Reset signal generation clears this register to 0000H.

**Figure 6-15. Format of Timer Channel Stop register m (TTm)**

Address: F01B4H, F01B5H (TT0) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TTm | 0 | 0 | 0 | 0 | TTHm3 | 0 | TTHm1 | 0 | TTm7 | TTm6 | TTm5 | TTm4 | TTm3 | TTm2 | TTm1 | TTm0 |

| TTHm3 | Trigger to stop operation of the higher 8-bit timer when channel 3 is in the 8-bit timer mode |
|---|---|
| 0 | No trigger operation |
| 1 | TEHm3 bit clear to 0, to be count operation stop status. |

| TTHm1 | Trigger to stop operation of the higher 8-bit timer when channel 1 is in the 8-bit timer mode |
|---|---|
| 0 | No trigger operation |
| 1 | TEHm1 bit clear to 0, to be count operation stop status. |

| TTmn | Operation stop trigger of channel n |
|---|---|
| 0 | No trigger operation |
| 1 | TEmn bit clear to 0, to be count operation stop status.<br>This bit is the trigger to stop operation of the lower 8-bit timer for TTm1 and TTm3 when channel 1 or 3 is in the 8-bit timer mode. |

**Caution Be sure to clear bits 15 to 12, 10, 8 of the TTm register to "0".**

**Remarks 1.** When the TTm register is read, 0 is always read.

**2.** m: Unit number (m = 0),n: Channel number (n = 0 to 7)

### 6.3.8 Timer input select register 0 (TIS0)

The TIS0 register is used to select the channel 5 of unit 0 timer input..

The TIS0 register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 6-16. Format of Timer Input Select register 0 (TIS0)**

Address: F0074H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TIS0 | 0 | 0 | 0 | 0 | 0 | TIS02 | TIS01 | TIS00 |

| TIS02 | TIS01 | TIS00 | Selection of timer input used with channel 5 |
|---|---|---|---|
| 0 | 0 | 0 | Input signal of timer input pin (TI05) |
| 0 | 0 | 1 | |
| 0 | 1 | 0 | |
| 0 | 1 | 1 | |
| 1 | 0 | 0 | Low-speed on-chip oscillator clock ($f_{IL}$) |
| 1 | 0 | 1 | Subsystem clock ($f_{SUB}$) |
| Other than above | | | Setting prohibited |

**Caution High-level width, low-level width of timer input is selected, will require more than 1/$f_{MCK}$ +10 ns. Therefore, when selecting $f_{SUB}$ to $f_{CLK}$ (CSS bit of CKS register = 1), can not TIS02 bit set to 1.**

### 6.3.9 Timer output enable register m (TOEm)

The TOEm register is used to enable or disable timer output of each channel.

Channel n for which timer output has been enabled becomes unable to rewrite the value of the TOmn bit of timer output register m (TOm) described later by software, and the value reflecting the setting of the timer output function through the count operation is output from the timer output pin (TOmn).

The TOEm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the TOEm register can be set with a 1-bit or 8-bit memory manipulation instruction with TOEmL.

Reset signal generation clears this register to 0000H.

**Figure 6-17. Format of Timer Output Enable register m (TOEm)**

Address: F01BAH, F01BBH (TOE0) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TOEm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | TOEm7 | TOEm6 | TOEm5 | TOEm4 | TOEm3 | 0 | TOEm1 | TOEm0 |

| TOEmn | Timer output enable/disable of channel n |
|---|---|
| 0 | Disable output of timer.<br>Without reflecting on TOmn bit timer operation, to fixed the output.<br>Writing to the TOmn bit is enabled, and level set to TOmn bit is output from the TOmn pin. |
| 1 | Enable output of timer.<br>Reflected in the TOmn bit timer operation, to generate the output waveform.<br>Writing to the TOmn bit is disabled (writing is ignored). |

**Caution Be sure to clear bits 15 to 8 to "0".**

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

### 6.3.10 Timer output register m (TOm)

The TOm register is a buffer register of timer output of each channel.

The value of each bit in this register is output from the timer output pin (TOmn) of each channel.

The TOmn bit oh this register can be rewritten by software only when timer output is disabled (TOEmn = 0). When timer output is enabled (TOEmn = 1), rewriting this register by software is ignored, and the value is changed only by the timer operation.

To use the P00/TI00, P01/TO00, P16/TI01/TO01, P31/TI03/TO03, P42/TI04/TO04, P05/TI05/TO05, P06/TI06/TO06, P41/TI07/TO07 pin as a port function pin, set the corresponding TOmn bit to “0”.

The TOm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the TOm register can be set with an 8-bit memory manipulation instruction with TOmL.

Reset signal generation clears this register to 0000H.

**Figure 6-18. Format of Timer Output register m (TOm)**

Address: F01B8H, F01B9H (TO0) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TOm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | TOm7 | TOm6 | TOm5 | TOm4 | TOm3 | 0 | TOm1 | TOm0 |

| TOmn | Timer output of channel n |
|---|---|
| 0 | Timer output value is “0”. |
| 1 | Timer output value is “1”. |

**Caution Be sure to clear bits 15 to 8 to “0”.**

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

### 6.3.11 Timer output level register m (TOLm)

The TOLm register is a register that controls the timer output level of each channel.

The setting of the inverted output of channel n by this register is reflected at the timing of set or reset of the timer output signal while the timer output is enabled (TOEmn = 1) in the Slave channel output mode (TOMmn = 1). In the master channel output mode (TOMmn = 0), this register setting is invalid.

The TOLm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the TOLm register can be set with an 8-bit memory manipulation instruction with TOLmL.

Reset signal generation clears this register to 0000H.

**Figure 6-19. Format of Timer Output Level register m (TOLm)**

Address: F01BCH, F01BDH (TOL0) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TOLm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | TOLm7 | TOLm6 | TOLm5 | TOLm4 | TOLm3 | 0 | TOLm1 | 0 |

| TOLmn | Control of timer output level of channel n |
|---|---|
| 0 | Positive logic output (active-high) |
| 1 | Negative logic output (active-low) |

**Caution Be sure to clear bits 15 to 8, and 0 to "0".**

**Remarks 1.** If the value of this register is rewritten during timer operation, the timer output logic is inverted when the timer output signal changes next, instead of immediately after the register value is rewritten.

**2.** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

### 6.3.12 Timer output mode register m (TOMm)

The TOMm register is used to control the timer output mode of each channel.

When a channel is used for the independent channel operation function, set the corresponding bit of the channel to be used to 0.

When a channel is used for the simultaneous channel operation function (PWM output, one-shot pulse output, or multiple PWM output), set the corresponding bit of the master channel to 0 and the corresponding bit of the slave channel to 1.

The setting of each channel n by this register is reflected at the timing when the timer output signal is set or reset while the timer output is enabled (TOEmn = 1).

The TOMm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the TOMm register can be set with an 8-bit memory manipulation instruction with TOMmL.

Reset signal generation clears this register to 0000H.

**Figure 6-20. Format of Timer Output Mode register m (TOMm)**

Address: F01BEH, F01BFH (TOM0) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TOMm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | TOMm7 | TOMm6 | TOMm5 | TOMm4 | TOMm3 | 0 | TOMm1 | 0 |

| TOMmn | Control of timer output mode of channel n |
|---|---|
| 0 | Master channel output mode (to produce toggle output by timer interrupt request signal (INTTMmn)) |
| 1 | Slave channel output mode (output is set by the timer interrupt request signal (INTTMmn) of the master channel, and reset by the timer interrupt request signal (INTTM0p) of the slave channel) |

**Caution Be sure to clear bits 15 to 8, and 0 to "0".**

**Remark** m: Unit number (m = 0)
n: Channel number
n = 0 to 7 (n = 0, 2, 4, 6 for master channel)
p: Slave channel number
$n < p \leq 7$
(For details of the relation between the master channel and slave channel, see **6.4.1 Basic rules of simultaneous channel operation function**.)

### 6.3.13 Input switch control register (ISC)

The ISC1 and ISC0 bits of the ISC register are used to implement LIN-bus communication operation by using channel 7 in association with the serial array unit. When the ISC1 bit is set to 1, the input signal of the serial data input pin (RxD2) is selected as a timer input signal.

The ISC register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 6-21. Format of Input Switch Control Register (ISC)**

Address: F0073H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ISC | 0 | 0 | 0 | 0 | 0 | 0 | ISC1 | ISC0 |

| ISC1 | Switching channel 7 input of timer array unit |
|---|---|
| <R> 0 | 48 and 64-pin products:<br>Uses the input signal of the TI07 pin as a timer input (normal operation).<br><R> 25 and 32-pin products:<br>Do not use a timer input signal for channel 7. |
| 1 | Input signal of the RxD2 pin is used as timer input (detects the wakeup signal and measures the low width of the break field and the pulse width of the sync field).<br>Setting is prohibited in the 25-pin products. |

| ISC0 | Switching external interrupt (INTP0) input |
|---|---|
| 0 | Uses the input signal of the INTP0 pin as an external interrupt (normal operation). |
| 1 | Uses the input signal of the RxD2 pin as an external interrupt (wakeup signal detection). |

**Caution Be sure to clear bits 7 to 2 to "0".**

**Remark** When the LIN-bus communication function is used, select the input signal of the RxD2 pin by setting ISC1 to 1.

### 6.3.14 Noise filter enable register 1 (NFEN1)

The NFEN1 register is used to set whether the noise filter can be used for the timer input signal to each channel.

Enable the noise filter by setting the corresponding bits to 1 on the pins in need of noise removal.

<R> When the noise filter is enabled, after synchronization with the operating clock ($f_{MCK}$) for the target channel, whether the signal keeps the same value for two clock cycles is detected. When the noise filter is disabled, the input signal is only synchronized with the operating clock ($f_{MCK}$) for the target channel[Note].

The NFEN1 registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Note** For details, see **6.5.1 (2) When valid edge of input signal via the TImn pin is selected (CCSmn = 1)**, **6.5.2 Start timing of counter** and **6.7 Timer Input (TImn) Cntorol**.

**Figure 6-22. Format of Noise Filter Enable Register 1 (NFEN1)**

Address: F0071H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| NFEN1 | TNFEN07 | TNFEN06 | TNFEN05 | TNFEN04 | TNFEN03 | 0 | TNFEN01 | TNFEN00 |

| TNFEN07 | Enable/disable using noise filter of TI07/TO07/P41 pin or RxD2/P14 pin input signal[Note] |
|---|---|
| 0 | Noise filter OFF |
| 1 | Noise filter ON |

| TNFEN06 | Enable/disable using noise filter of TI06/TO06/P06 pin input signal |
|---|---|
| 0 | Noise filter OFF |
| 1 | Noise filter ON |

| TNFEN05 | Enable/disable using noise filter of TI05/TO05/P05 pin input signal |
|---|---|
| 0 | Noise filter OFF |
| 1 | Noise filter ON |

| TNFEN04 | Enable/disable using noise filter of TI04/TO04/P42 pin input signal |
|---|---|
| 0 | Noise filter OFF |
| 1 | Noise filter ON |

| TNFEN03 | Enable/disable using noise filter of TI03/TO03/P31 pin input signal |
|---|---|
| 0 | Noise filter OFF |
| 1 | Noise filter ON |

| TNFEN01 | Enable/disable using noise filter of TI01/TO01/P16 pin input signal |
|---|---|
| 0 | Noise filter OFF |
| 1 | Noise filter ON |

| TNFEN00 | Enable/disable using noise filter of TI00/P00 pin input signal |
|---|---|
| 0 | Noise filter OFF |
| 1 | Noise filter ON |

**Note** The applicable pin can be switched by setting the ISC1 bit of the ISC register.

ISC1 = 0: Whether or not to use the noise filter of the TI07 pin can be selected.

ISC1 = 1: Whether or not to use the noise filter of the RxD2 pin can be selected.

**Caution Be sure to clear bit 2 to "0".**

**Remark** The presence or absence of timer I/O pins of channel 0 to 7 depends on the product. See **Table 6-2 Timer I/O Pins provided in Each Product** for details.

<R> **6.3.15 Registers controlling port functions of pins to be used for timer I/O**

Using port pins for the timer array unit functions requires setting of the registers that control the port functions multiplexed on the target pins (port mode register (PMxx), port register (Pxx), and port mode control register (PMCxx)). For details, see **4.3.1 Port mode registers (PMxx)**, **4.3.2 Port registers (Pxx)**, and **4.3.6 Port mode control registers (PMCxx)**.

The port mode register (PMxx), port register (Pxx), and port mode control register (PMCxx) to be set depend on the product. For details, see **4.5.3 Register setting examples for used port and alternate functions**.

When using the ports (such as P31/TI03/TO03) to be shared with the timer output pin for timer output, set the port mode control register (PMCxx) bit, port mode register (PMxx) bit and port register (Pxx) bit corresponding to each port to 0.

Example: When using P31/TI03/TO03 for timer output
Set the PMC31 bit of port mode control register 3 to 0.
Set the PM31 bit of port mode register 3 to 0.
Set the P31 bit of port register 3 to 0.

When using the ports (such as P31/TI03/TO03) to be shared with the timer input pin for timer input, set the port mode register (PMxx) bit corresponding to each port to 1. And set the port mode control register (PMCxx) bit corresponding to each port to 0. At this time, the port register (Pxx) bit may be 0 or 1.

Example: When using P31/TI03/TO03 for timer input
Set the PMC31 bit of port mode control register 3 to 0.
Set the PM31 bit of port mode register 3 to 1.
Set the P31 bit of port register 3 to 0 or 1.

## 6.4 Basic Rules of Timer Array Unit

### 6.4.1 Basic rules of simultaneous channel operation function

When simultaneously using multiple channels, namely, a combination of a master channel (a reference timer mainly counting the cycle) and slave channels (timers operating according to the master channel), the following rules apply.

(1) Only an even channel (channel 0, 2, 4, etc.) can be set as a master channel.
(2) Any channel, except channel 0, can be set as a slave channel.
(3) The slave channel must be lower than the master channel.
Example: If channel 2 is set as a master channel, channel 3 or those that follow (channels 3, 4, 5, etc.) can be set as a slave channel.

(4) Two or more slave channels can be set for one master channel.
(5) When two or more master channels are to be used, slave channels with a master channel between them may not be set.
Example: If channels 0 and 4 are set as master channels, channels 1 to 3 can be set as the slave channels of master channel 0. Channels 5 to 7 cannot be set as the slave channels of master channel 0.

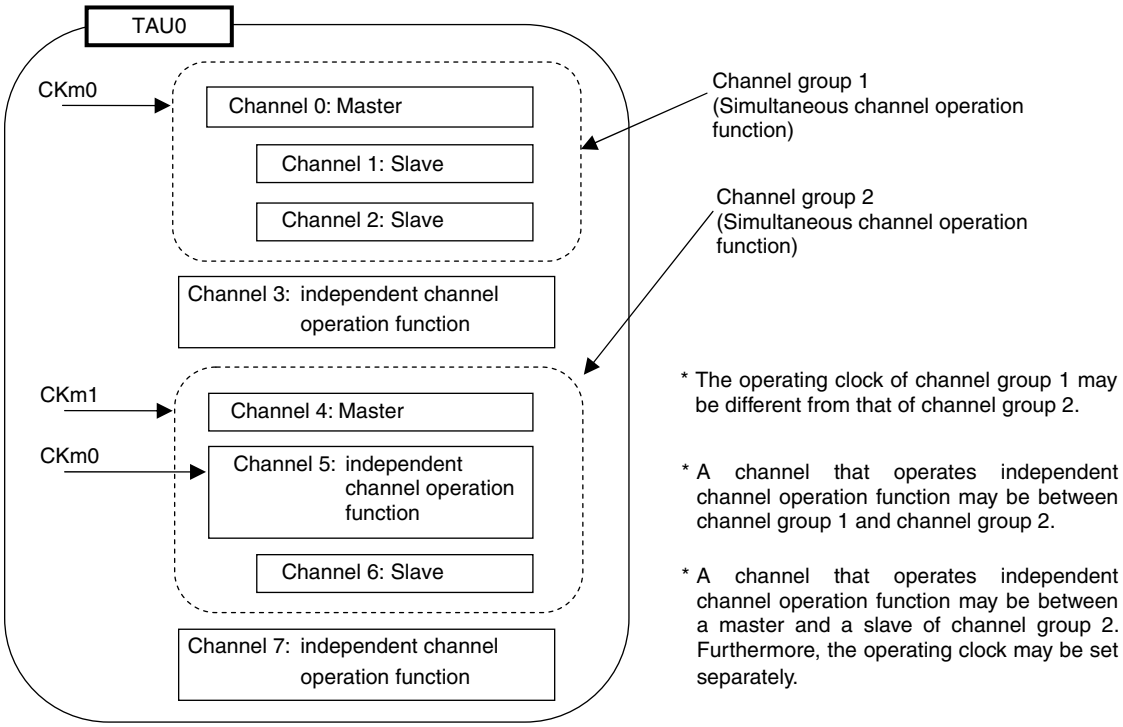
(6) The operating clock for a slave channel in combination with a master channel must be the same as that of the master channel. The CKSmn0, CKSmn1 bits (bit 15, 14 of timer mode register mn (TMRmn)) of the slave channel that operates in combination with the master channel must be the same value as that of the master channel.
(7) A master channel can transmit INTTMmn (interrupt), start software trigger, and count clock to the lower channels.
(8) A slave channel can use INTTMmn (interrupt), a start software trigger, or the count clock of the master channel as a source clock, but cannot transmit its own INTTMmn (interrupt), start software trigger, or count clock to channels with lower channel numbers.
(9) A master channel cannot use INTTMmn (interrupt), a start software trigger, or the count clock from the other higher master channel as a source clock.
(10) To simultaneously start channels that operate in combination, the channel start trigger bit (TSmn) of the channels in combination must be set at the same time.
(11) During the counting operation, a TSmn bit of a master channel or TSmn bits of all channels which are operating simultaneously can be set. It cannot be applied to TSmn bits of slave channels alone.
(12) To stop the channels in combination simultaneously, the channel stop trigger bit (TTmn) of the channels in combination must be set at the same time.
(13) CKm2/CKm3 cannot be selected while channels are operating simultaneously, because the operating clocks of master channels and slave channels have to be synchronized.
(14) Timer mode register m0 (TMRm0) has no master bit (it is fixed as “0”). However, as channel 0 is the highest channel, it can be used as a master channel during simultaneous operation.

The rules of the simultaneous channel operation function are applied in a channel group (a master channel and slave channels forming one simultaneous channel operation function).

If two or more channel groups that do not operate in combination are specified, the basic rules of the simultaneous channel operation function in **6.4.1 Basic rules of simultaneous channel operation function** do not apply to the channel groups.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

Example




* The operating clock of channel group 1 may be different from that of channel group 2.

* A channel that operates independent channel operation function may be between channel group 1 and channel group 2.

* A channel that operates independent channel operation function may be between a master and a slave of channel group 2. Furthermore, the operating clock may be set separately.

### 6.4.2 Basic rules of 8-bit timer operation function (channels 1 and 3 only)

The 8-bit timer operation function makes it possible to use a 16-bit timer channel in a configuration consisting of two 8-bit timer channels.

This function can only be used for channels 1 and 3, and there are several rules for using it.

The basic rules for this function are as follows:

(1) The 8-bit timer operation function applies only to channels 1 and 3.
(2) When using 8-bit timers, set the SPLIT bit of timer mode register mn (TMRmn) to 1.
(3) The higher 8 bits can be operated as the interval timer function.
(4) At the start of operation, the higher 8 bits output INTTMm1H/INTTMm3H (an interrupt) (which is the same operation performed when MDmn0 is set to 1).
(5) The operation clock of the higher 8 bits is selected according to the CKSmn1 and CKSmn0 bits of the lower-bit TMRmn register.
(6) For the higher 8 bits, the TSHm1/TSHm3 bit is manipulated to start channel operation and the TTHm1/TTHm3 bit is manipulated to stop channel operation. The channel status can be checked using the TEHm1/TEHm3 bit.
(7) The lower 8 bits operate according to the TMRmn register settings. The following three functions support operation of the lower 8 bits:
- Interval timer function
- External event counter function
- Delay count function

(8) For the lower 8 bits, the TSm1/TSm3 bit is manipulated to start channel operation and the TTm1/TTm3 bit is manipulated to stop channel operation. The channel status can be checked using the TEm1/TEm3 bit.
(9) During 16-bit operation, manipulating the TSHm1, TSHm3, TTHm1, and TTHm3 bits is invalid. The TSm1, TSm3, TTm1, and TTm3 bits are manipulated to operate channels 1 and 3. The TEHm3 and TEHm1 bits are not changed.
(10) For the 8-bit timer function, the simultaneous operation functions (one-shot pulse, PWM, and multiple PWM) cannot be used.

**Remark** m: Unit number (m = 0), n: Channel number (n = 1, 3)

## 6.5 Operation of Counter

### 6.5.1 Count clock ($f_{TCLK}$)

The count clock ($f_{TCLK}$) of the timer array unit can be selected between following by CCSmn bit of timer mode register mn (TMRmn). .

- Operation clock ($f_{MCK}$) specified by the CKSmn0 and CKSmn1 bits
- Valid edge of input signal input from the TImn pin

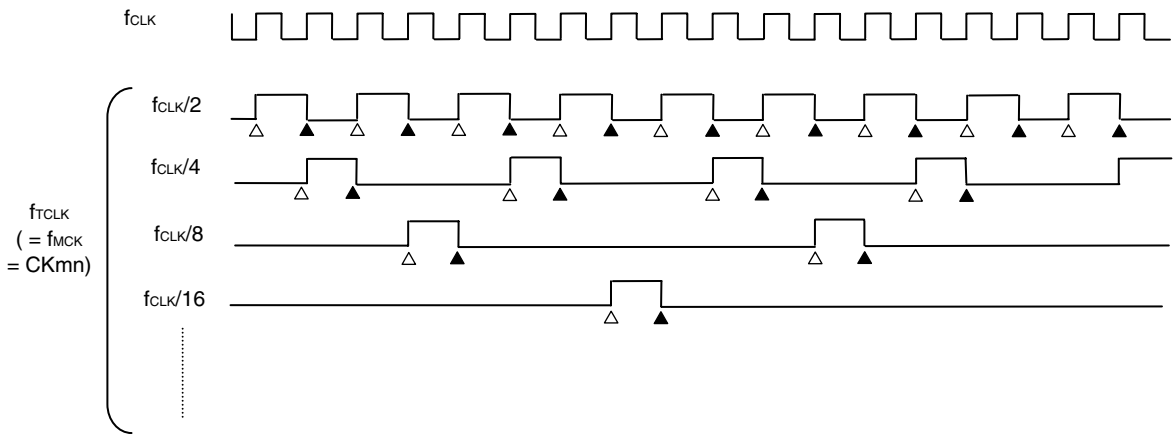
Because the timer array unit is designed to operate in synchronization with $f_{CLK}$, the timings of the count clock ($f_{TCLK}$) are shown below.

**(1) When operation clock ($f_{MCK}$) specified by the CKSmn0 and CKSmn1 bits is selected (CCSmn = 0)**

The count clock ($f_{TCLK}$) is between $f_{CLK}$ to $f_{CLK}/2^{15}$ by setting of timer clock select register m (TPSm). When a divided $f_{CLK}$ is selected, however, the clock selected in TPSmn register, but a signal which becomes high level for one period of $f_{CLK}$ from its rising edge. When a $f_{CLK}$ is selected, fixed to high level.

Counting of timer count register mn (TCRmn) delayed by one period of $f_{CLK}$ from rising edge of the count clock, because of synchronization with $f_{CLK}$. But, this is described as “counting at rising edge of the count clock”, as a matter of convenience.

**Figure 6-23. Timing of $f_{CLK}$ and count clock ($f_{TCLK}$) (When CCSmn = 0)**




**Remarks 1.** △ : Rising edge of the count clock
▲ : Synchronization, increment/decrement of counter

**2.** $f_{CLK}$: CPU/peripheral hardware clock

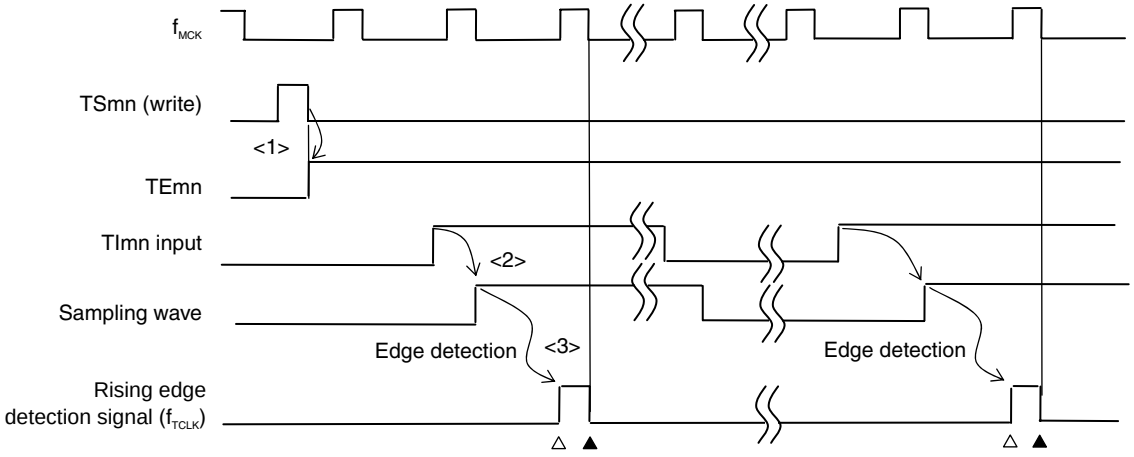
**3.** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

**(2) When valid edge of input signal via the TImn pin is selected (CCSmn = 1)**

The count clock ($f_{TCLK}$) becomes the signal that detects valid edge of input signal via the TImn pin and synchronizes next rising $f_{MCK}$. The count clock ($f_{TCLK}$) is delayed for 1 to 2 period of $f_{MCK}$ from the input signal via the TImn pin (when a noise filter is used, the delay becomes 3 to 4 clock).

Counting of timer count register mn (TCRmn) delayed by one period of $f_{CLK}$ from rising edge of the count clock, because of synchronization with $f_{CLK}$. But, this is described as "counting at valid edge of input signal via the TImn pin", as a matter of convenience.

**Figure 6-24. Timing of $f_{CLK}$ and count clock ($f_{TCLK}$) (When CCSmn = 1, noise filter unused)**




<1> Setting TSmn bit to 1 enables the timer to be started and to become wait state for valid edge of input signal via the TImn pin.

<2> The rise of input signal via the TImn pin is sampled by $f_{MCK}$.

<3> The edge is detected by the rising of the sampled signal and the detection signal (count clock) is output.

**Remarks 1.** △ : Rising edge of the count clock
▲ : Synchronization, increment/decrement of counter

**2.** $f_{CLK}$: CPU/peripheral hardware clock
$f_{MCK}$: Operation clock of channel n

**3.** The waveform of the input signal via TImn pin of the input pulse interval measurement, the measurement of high/low width of input signal, and the delay counter, the one-shot pulse output are the same as that shown in Figure 6-22.

**4.** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

### 6.5.2 Start timing of counter

Operation of timer count register mn (TCRmn) is enabled by setting of TSmn bit of timer channel start register m (TSm). Operation from when counting is enabled to when timer count register mn (TCRmn) starts counting is shown in Table 6-6.

**Table 6-6. Operations from Count Operation Enabled State to Timer count Register mn (TCRmn) Count Start**

| Timer operation mode | Operation when TSmn = 1 is set |
|---|---|
| • Interval timer mode | No operation is carried out from start trigger detection (TSmn=1) until count clock generation.<br>The first count clock loads the value of the TDRmn register to the TCRmn register and the subsequent count clock performs count down operation (see **6.5.3 (1) Operation of interval timer mode**). |
| • Event counter mode | Writing 1 to the TSmn bit loads the value of the TDRmn register to the TCRmn register.<br>If detect edge of TImn input. The subsequent count clock performs count down operation (see **6.5.3 (2) Operation of event counter mode**). |
| • Capture mode | No operation is carried out from start trigger detection (TSmn = 1) until count clock generation.<br>The first count clock loads 0000H to the TCRmn register and the subsequent count clock performs count up operation (see **6.5.3 (3) Operation of capture mode (input pulse interval measurement)**). |
| • One-count mode | The waiting-for-start-trigger state is entered by writing 1 to the TSmn bit while the timer is stopped (TEmn = 0).<br>No operation is carried out from start trigger detection until count clock generation.<br>The first count clock loads the value of the TDRmn register to the TCRmn register and the subsequent count clock performs count down operation (see **6.5.3 (4) Operation of one-count mode**). |
| • Capture & one-count mode | The waiting-for-start-trigger state is entered by writing 1 to the TSmn bit while the timer is stopped (TEmn = 0).<br>No operation is carried out from start trigger detection until count clock generation.<br>The first count clock loads 0000H to the TCRmn register and the subsequent count clock performs count up operation (see **6.5.3 (5) Operation of capture & one-count mode (high-level interval measurement)**). |

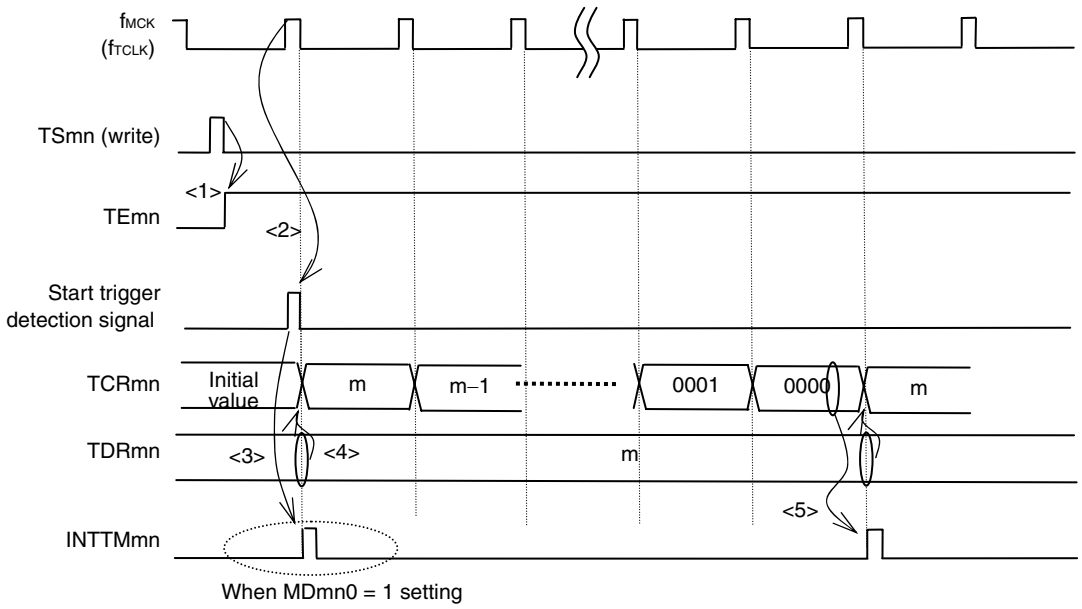
### 6.5.3 Operation of counter

Here, the counter operation in each mode is explained.

#### (1) Operation of interval timer mode

<1> Operation is enabled (TEmn = 1) by writing 1 to the TSmn bit. Timer count register mn (TCRmn) holds the initial value until count clock generation.
<2> A start trigger is generated at the first count clock after operation is enabled.
<3> When the MDmn0 bit is set to 1, INTTMmn is generated by the start trigger.
<4> By the first count clock after the operation enable, the value of timer data register mn (TDRmn) is loaded to the TCRmn register and counting starts in the interval timer mode.
<5> When the TCRmn register counts down and its count value is 0000H, INTTMmn is generated and the value of timer data register mn (TDRmn) is loaded to the TCRmn register and counting keeps on.

**Figure 6-25. Operation Timing (In Interval Timer Mode)**




**Caution In the operation in the first count clock cycle after writing the TSmn bit, an error at a maximum of one count clock cycle occurs since count start delays until count clock has been generated. When the information on count start timing is necessary, an interrupt can be generated when counting is started by setting MDmn0 = 1.**

**Remarks 1.** fMCK, the start trigger detection signal, and INTTMmn become active between one clock in synchronization with fCLK.

**2.** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

### (2) Operation of event counter mode

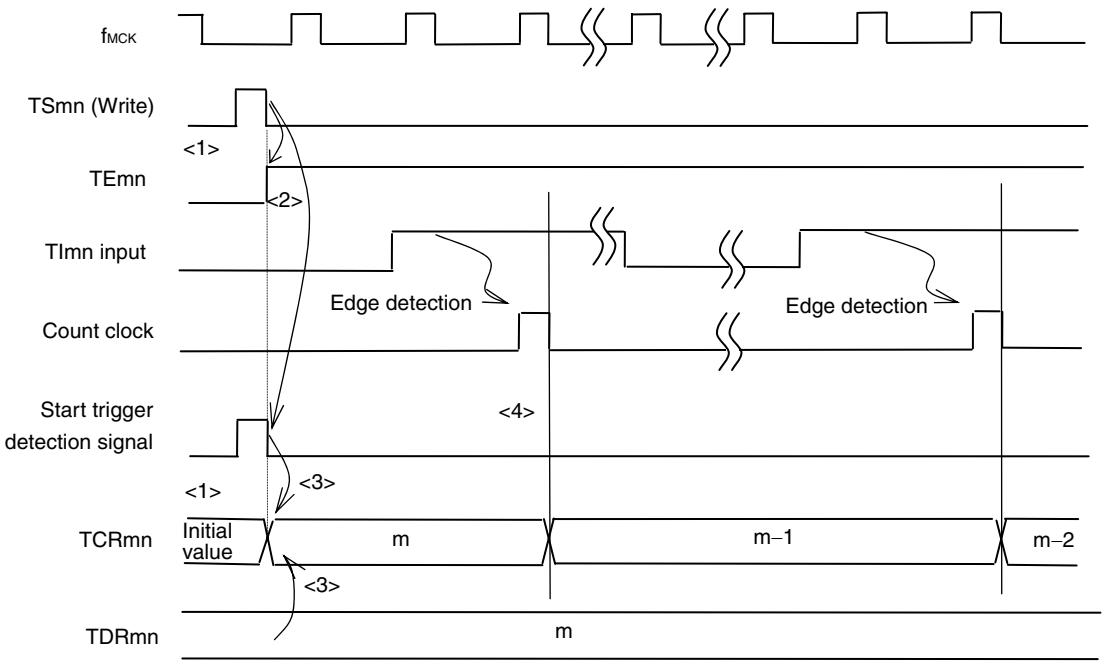
<1> Timer count register mn (TCRmn) holds its initial value while operation is stopped (TEmn = 0).

<2> Operation is enabled (TEmn = 1) by writing 1 to the TSmn bit.

<3> As soon as 1 has been written to the TSmn bit and 1 has been set to the TEmn bit, the value of timer data register mn (TDRmn) is loaded to the TCRmn register to start counting.

<4> After that, the TCRmn register value is counted down according to the count clock of the valid edge of the TImn input.

**Figure 6-26. Operation Timing (In Event Counter Mode)**



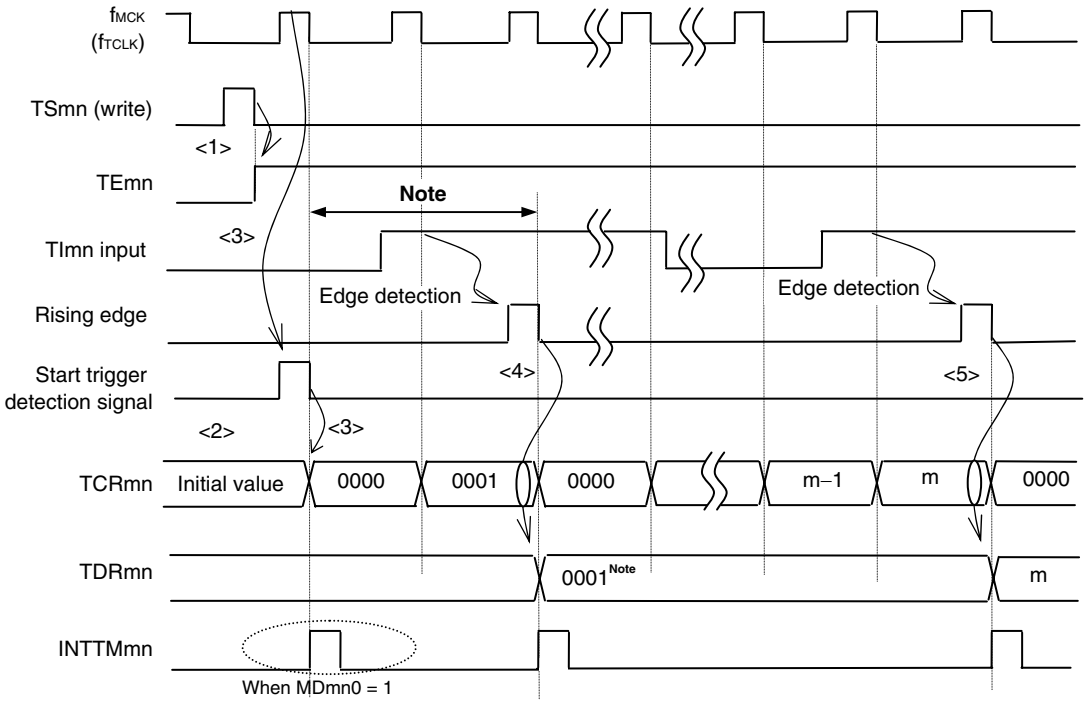

<R> **Remarks 1.** The timing is shown in Figure 6-24 indicates while the noise filter is not used. By making the noise filter on-state, the edge detection becomes 2 $f_{MCK}$ cycles (it sums up to 3 to 4 cycles) later than the normal cycle of TImn input. The error per one period occurs by the asynchronous between the period of the TImn input and that of the count clock ($f_{MCK}$).

**2.** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

### (3) Operation of capture mode (input pulse interval measurement)

<1> Operation is enabled (TEmn = 1) by writing 1 to the TSmn bit.
<2> Timer count register mn (TCRmn) holds the initial value until count clock generation.
<3> A start trigger is generated at the first count clock after operation is enabled. And the value of 0000H is loaded to the TCRmn register and counting starts in the capture mode. (When the MDmn0 bit is set to 1, INTTMmn is generated by the start trigger.)
<4> On detection of the valid edge of the TImn input, the value of the TCRmn register is captured to timer data register mn (TDRmn) and INTTMmn is generated. However, this capture value is no meaning. The TCRmn register keeps on counting from 0000H.
<5> On next detection of the valid edge of the TImn input, the value of the TCRmn register is captured to timer data register mn (TDRmn) and INTTMmn is generated.

**Figure 6-27. Operation Timing (In Capture Mode : Input Pulse Interval Measurement)**




**Note** If a clock has been input to TImn (the trigger exists) when capturing starts, counting starts when a trigger is detected, even if no edge is detected. Therefore, the first captured value (<4>) does not determine a pulse interval (in the above figure, 0001 just indicates two clock cycles but does not determine the pulse interval) and so the user can ignore it.

**Caution In the first cycle operation of count clock after writing the TSmn bit, an error at a maximum of one clock is generated since count start delays until count clock has been generated. When the information on count start timing is necessary, an interrupt can be generated at count start by setting MDmn0 = 1.**

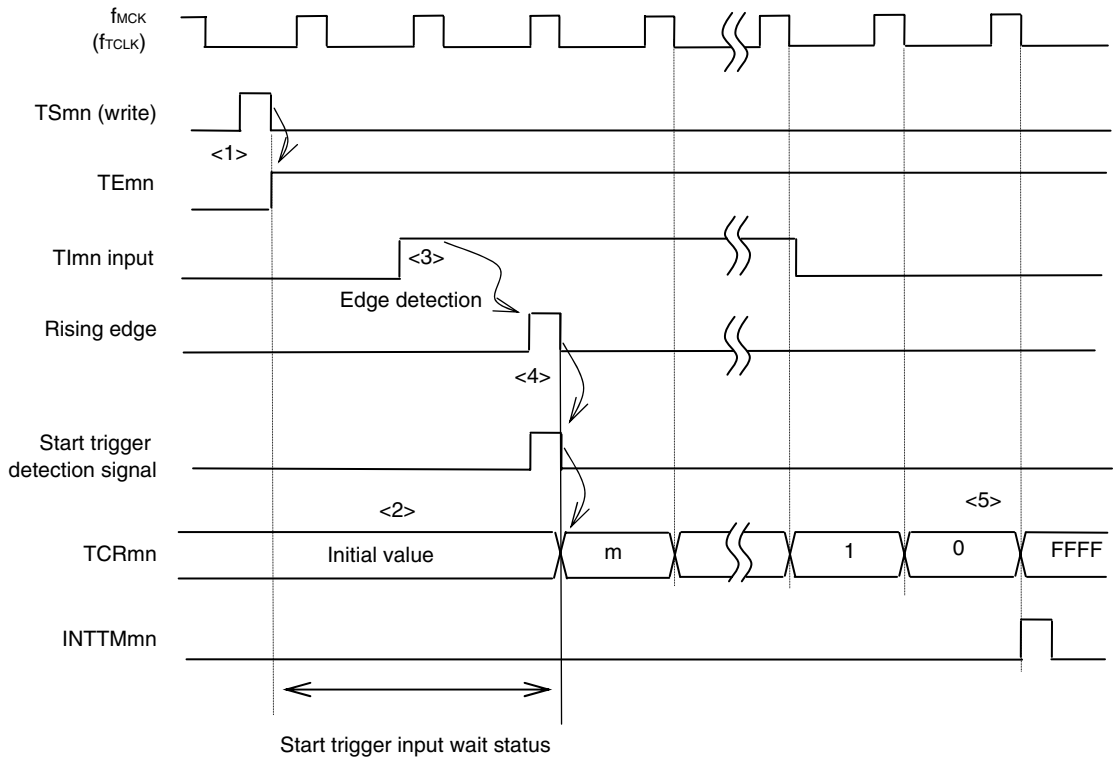
<R> **Remarks 1.** The timing is shown in Figure 6-25 indicates while the noise filter is not used. By making the noise filter on-state, the edge detection becomes 2 fMCK cycles (it sums up to 3 to 4 cycles) later than the normal cycle of TImn input. The error per one period occurs by the asynchronous between the period of the TImn input and that of the count clock (fMCK).

**2.** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

### (4) Operation of one-count mode

<1> Operation is enabled (TEmn = 1) by writing 1 to the TSmn bit.
<2> Timer count register mn (TCRmn) holds the initial value until start trigger generation.
<3> Rising edge of the TImn input is detected.
<4> On start trigger detection, the value of timer data register mn (TDRmn) is loaded to the TCRmn register and count starts.
<5> When the TCRmn register counts down and its count value is 0000H, INTTMmn is generated and the value of the TCRmn register becomes FFFFH and counting stops.

**Figure 6-28. Operation Timing (In One-count Mode)**



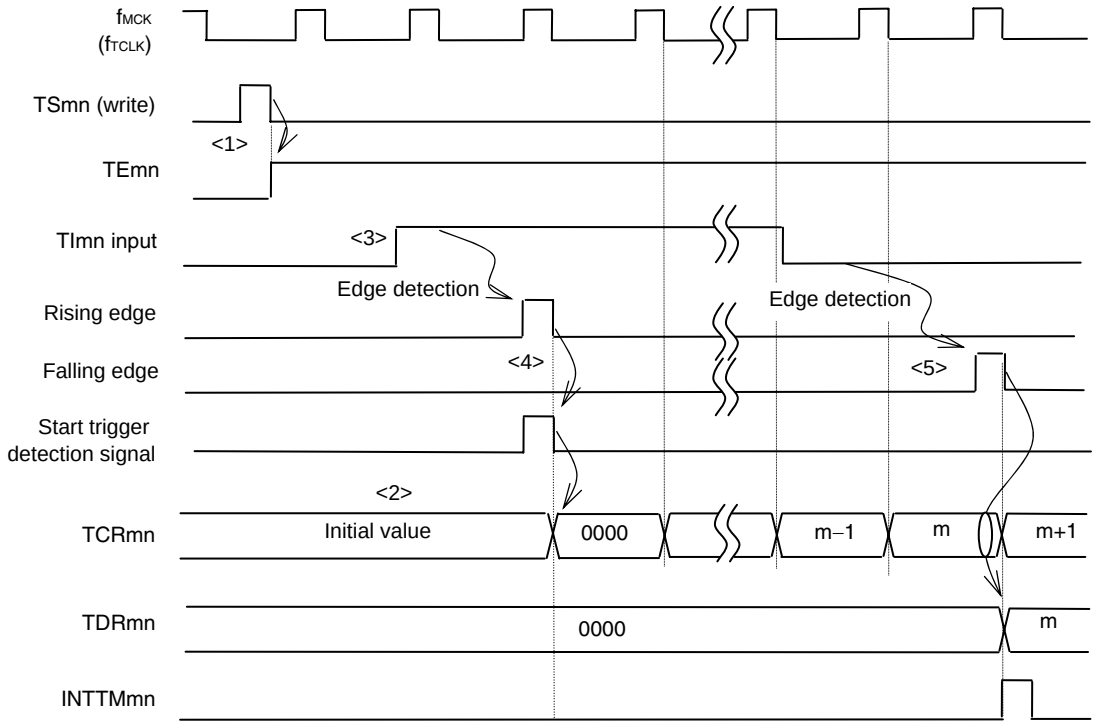

**Remarks 1.** The timing is shown in Figure 6-26 indicates while the noise filter is not used. By making the noise filter on-state, the edge detection becomes 2 $f_{MCK}$ cycles (it sums up to 3 to 4 cycles) later than the normal cycle of TImn input. The error per one period occurs by the asynchronous between the period of the TImn input and that of the count clock ($f_{MCK}$).

**2.** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

### (5) Operation of capture & one-count mode (high-level width measurement)

<1> Operation is enabled (TEmn = 1) by writing 1 to the TSmn bit of timer channel start register m (TSm).
<2> Timer count register mn (TCRmn) holds the initial value until start trigger generation.
<3> Rising edge of the TImn input is detected.
<4> On start trigger detection, the value of 0000H is loaded to the TCRmn register and count starts.
<5> On detection of the falling edge of the TImn input, the value of the TCRmn register is captured to timer data register mn (TDRmn) and INTTMmn is generated.

**Figure 6-29. Operation Timing (In Capture & One-count Mode : High-level Width Measurement)**



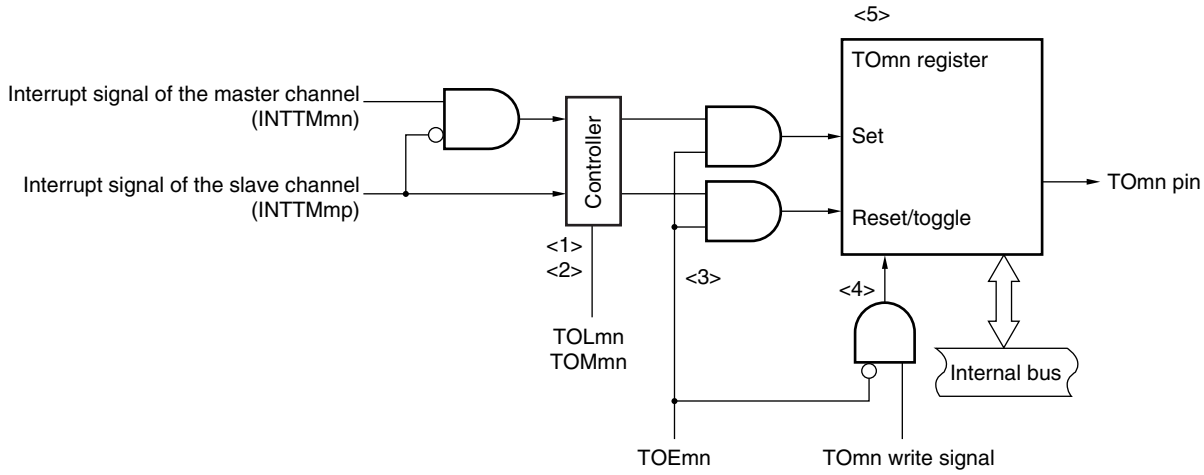

**Remarks 1.** The timing is shown in Figure 6-27 indicates while the noise filter is not used. By making the noise filter on-state, the edge detection becomes 2 fMCK cycles (it sums up to 3 to 4 cycles) later than the normal cycle of TImn input. The error per one period occurs by the asynchronous between the period of the TImn input and that of the count clock (fMCK).

**2.** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

## 6.6 Channel Output (TOmn Pin) Control

### 6.6.1 TOmn pin output circuit configuration

**Figure 6-30. Output Circuit Configuration**




The following describes the TOmn pin output circuit.

<1> When TOMmn = 0 (master channel output mode), the set value of timer output level register m (TOLm) is ignored and only INTTM0p (slave channel timer interrupt) is transmitted to timer output register m (TOm).

<2> When TOMmn = 1 (slave channel output mode), both INTTMmn (master channel timer interrupt) and INTTM0p (slave channel timer interrupt) are transmitted to the TOm register.
At this time, the TOLm register becomes valid and the signals are controlled as follows:

When TOLmn = 0: Positive logic output (INTTMmn → set, INTTM0p → reset)
When TOLmn = 1: Negative logic output (INTTMmn → reset, INTTM0p → set)

When INTTMmn and INTTM0p are simultaneously generated, (0% output of PWM), INTTM0p (reset signal) takes priority, and INTTMmn (set signal) is masked.

<3> While timer output is enabled (TOEmn = 1), INTTMmn (master channel timer interrupt) and INTTM0p (slave channel timer interrupt) are transmitted to the TOm register. Writing to the TOm register (TOmn write signal) becomes invalid.
When TOEmn = 1, the TOmn pin output never changes with signals other than interrupt signals.
To initialize the TOmn pin output level, it is necessary to set timer operation is stopped (TOEmn = 0) and to write a value to the TOm register.

<4> While timer output is disabled (TOEmn = 0), writing to the TOmn bit to the target channel (TOmn write signal) becomes valid. When timer output is disabled (TOEmn = 0), neither INTTMmn (master channel timer interrupt) nor INTTM0p (slave channel timer interrupt) is transmitted to the TOm register.

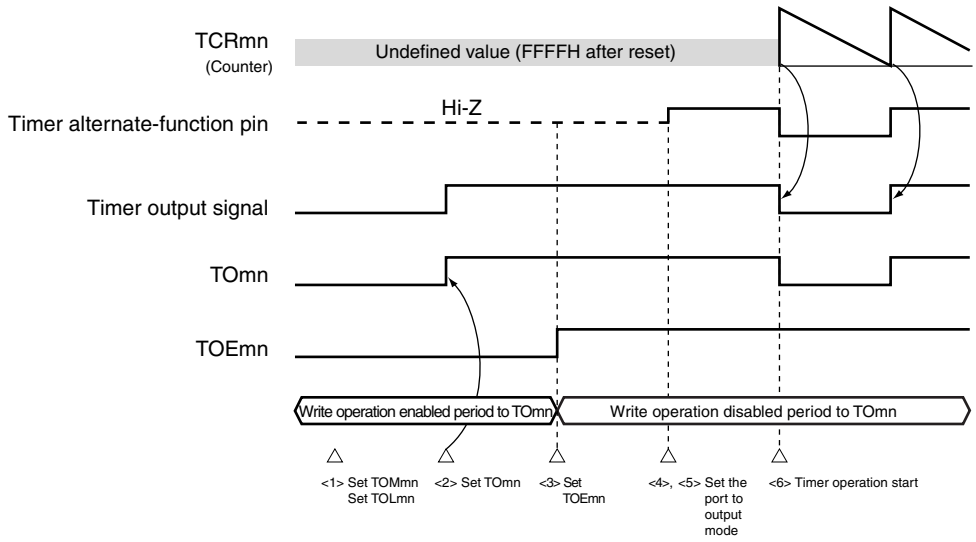
<5> The TOm register can always be read, and the TOmn pin output level can be checked.

**Remark** m: Unit number (m = 0)
n: Channel number
n = 0, 1, 3 to 7
p: Slave channel number
$n < p \leq 7$

### 6.6.2 TOmn pin output setting

The following figure shows the procedure and status transition of the TOmn output pin from initial setting to timer operation start.

**Figure 6-31. Status Transition from Timer Output Setting to Operation Start**




<1> The operation mode of timer output is set.

- TOMmn bit (0: Master channel output mode, 1: Slave channel output mode)
- TOLmn bit (0: Positive logic output, 1: Negative logic output)

<2> The timer output signal is set to the initial status by setting timer output register m (TOm).
<3> The timer output operation is enabled by writing 1 to the TOEmn bit (writing to the TOm register is disabled).
<4> The port is set to digital I/O by port mode control register (PMCxx) (see **6.3.15 Registers controlling port functions of pins to be used for timer I/O**).
<5> The port I/O setting is set to output (see **6.3.15 Registers controlling port functions of pins to be used for timer I/O**).
<6> The timer operation is enabled (TSmn = 1).

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 1, 3 to 7)

### 6.6.3 Cautions on channel output operation

**(1) Changing values set in the registers TOm, TOEm, and TOLm during timer operation**

Since the timer operations (operations of timer count register mn (TCRmn) and timer data register mn (TDRmn)) are independent of the TOmn output circuit and changing the values set in timer output register m (TOm), timer output enable register m (TOEm), and timer output level register m (TOLm) does not affect the timer operation, the values can be changed during timer operation. To output an expected waveform from the TOmn pin by timer operation, however, set the TOm, TOEm, TOLm, and TOMm registers to the values stated in the register setting example of each operation shown by 6.8 and 6.9.

When the values set to the TOEm, and TOLm registers (but not the TOm register) are changed close to the occurrence of the timer interrupt (INTTMmn) of each channel, the waveform output to the TOmn pin might differ, depending on whether the values are changed immediately before or immediately after the timer interrupt (INTTMmn) occurs.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 1, 3 to 7)

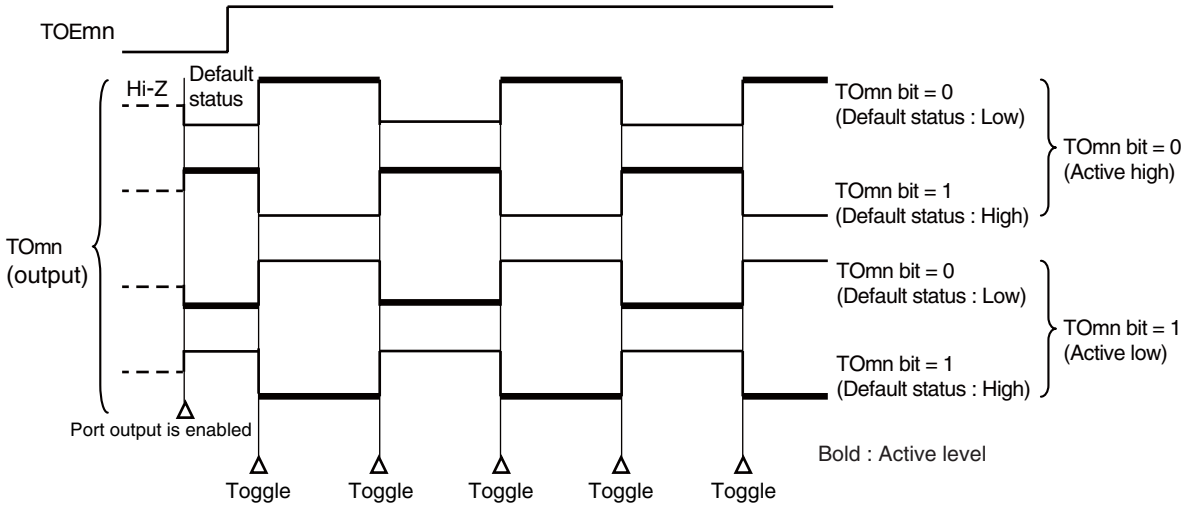
**(2) Default level of TOmn pin and output level after timer operation start**

The change in the output level of the TOmn pin when timer output register m (TOm) is written while timer output is disabled (TOEmn = 0), the initial level is changed, and then timer output is enabled (TOEmn = 1) before port output is enabled, is shown below.

**(a) When operation starts with master channel output mode (TOMmn = 0) setting**

The setting of timer output level register m (TOLm) is invalid when master channel output mode (TOMmn = 0). When the timer operation starts after setting the default level, the toggle signal is generated and the output level of the TOmn pin is reversed.

**Figure 6-32. TOmn Pin Output Status at Toggle Output (TOMmn = 0)**



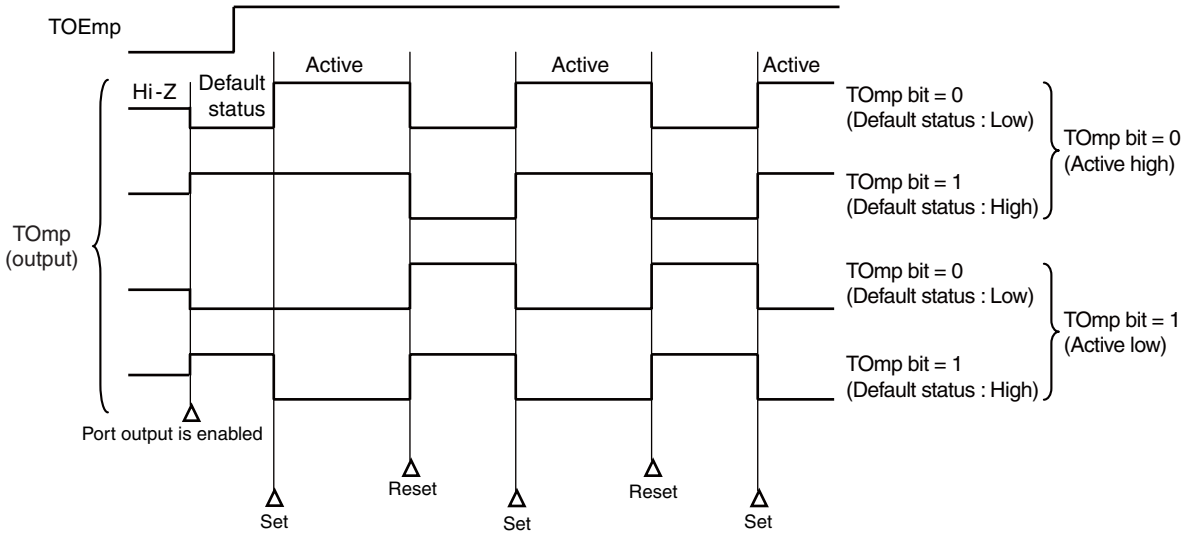

**Remarks 1.** Toggle: Reverse TOmn pin output status

**2.** m: Unit number (m = 0), n: Channel number (n = 0, 1, 3 to 7)

**(b) When operation starts with slave channel output mode (TOMmp = 1) setting (PWM output))**

When slave channel output mode (TOMmp = 1), the active level is determined by timer output level register m (TOLm) setting.

**Figure 6-33. TOmp Pin Output Status at PWM Output (TOMmp = 1)**




**Remarks 1.** Set: The output signal of the TOmp pin changes from inactive level to active level.
Reset: The output signal of the TOmp pin changes from active level to inactive level.

**2.** m: Unit number (m = 0), p: Channel number (p = 1, 3 to 7)

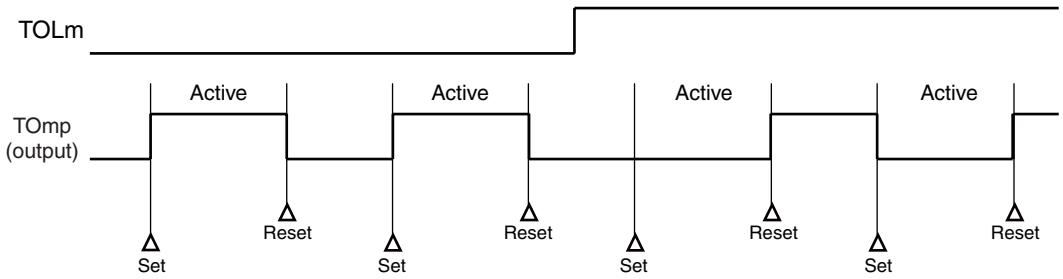
### (3) Operation of TOmn pin in slave channel output mode (TOMmn = 1)

#### (a) When timer output level register m (TOLm) setting has been changed during timer operation

When the TOLm register setting has been changed during timer operation, the setting becomes valid at the generation timing of the TOmn pin change condition. Rewriting the TOLm register does not change the output level of the TOmn pin.

The operation when TOMmn is set to 1 and the value of the TOLm register is changed while the timer is operating (TEmn = 1) is shown below.

**Figure 6-34. Operation when TOLm Register Has Been Changed Contents during Timer Operation**




**Remarks 1.** Set: The output signal of the TOmn pin changes from inactive level to active level.
Reset: The output signal of the TOmn pin changes from active level to inactive level.

**2.** m: Unit number (m = 0), n: Channel number (n = 0, 1, 3 to 7)

#### (b) Set/reset timing

To realize 0%/100% output at PWM output, the TOmn pin/TOmn bit set timing at master channel timer interrupt (INTTMmn) generation is delayed by 1 count clock by the slave channel.

If the set condition and reset condition are generated at the same time, a higher priority is given to the latter.

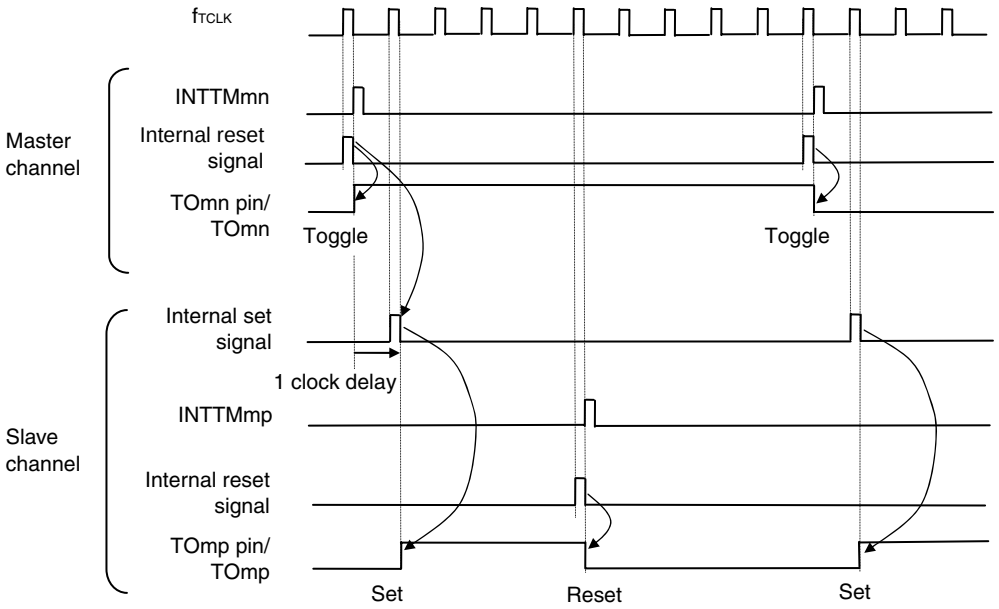
Figure 6-35 shows the set/reset operating statuses where the master/slave channels are set as follows.

Master channel: TOEmn = 1, TOMmn = 0, TOLmn = 0
Slave channel: TOEmp = 1, TOMmp = 1, TOLmp = 0

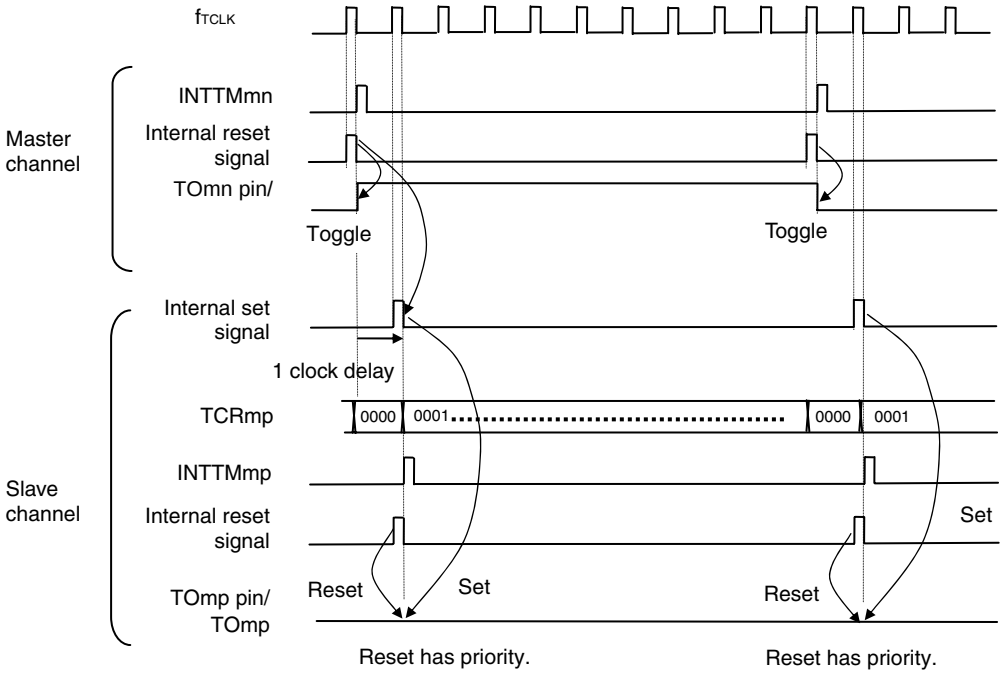
**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 1, 3 to 7)

**Figure 6-35. Set/Reset Timing Operating Statuses**

**(1) Basic operation timing**




**(2) Operation timing when 0 % duty**




**Remarks 1.** Internal reset signal: TOmn pin reset/toggle signal
Internal set signal: TOmn pin set signal

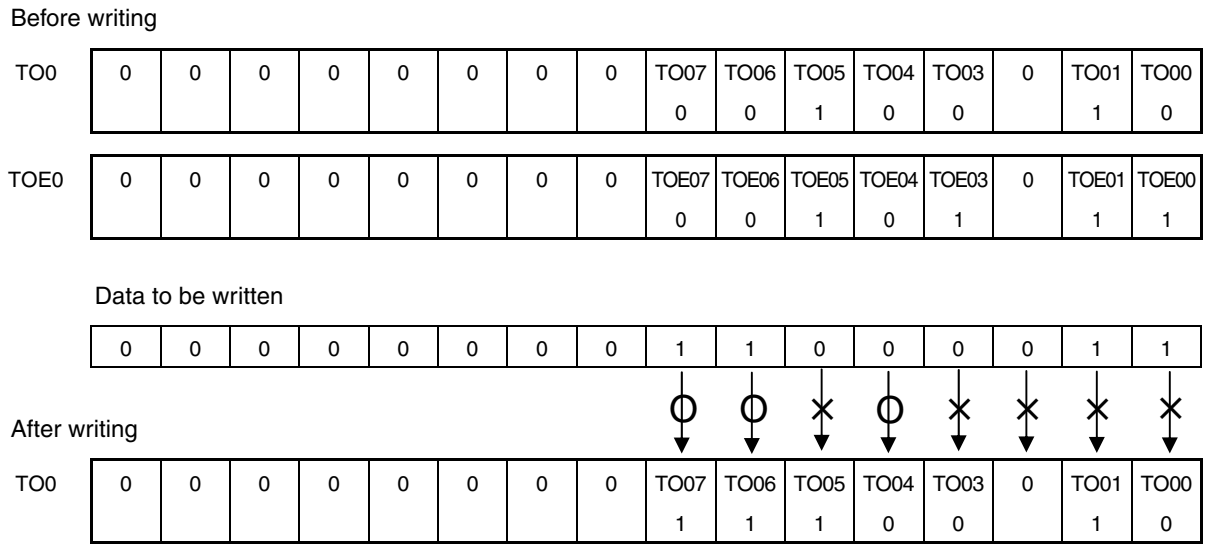
**2.** m: Unit number (m = 0)
n: Channel number
n = 0, 1, 3 to 7 (n = 0, 4, 6 for master channel)
p: Slave channel number
$n < p \leq 7$

### 6.6.4 Collective manipulation of TOmn bit

In timer output register m (TOm), the setting bits for all the channels are located in one register in the same way as timer channel start register m (TSm). Therefore, the TOmn bit of all the channels can be manipulated collectively.

Only the desired bits can also be manipulated by enabling writing only to the TOmn bits (TOEmn = 0) that correspond to the relevant bits of the channel used to perform output (TOmn).

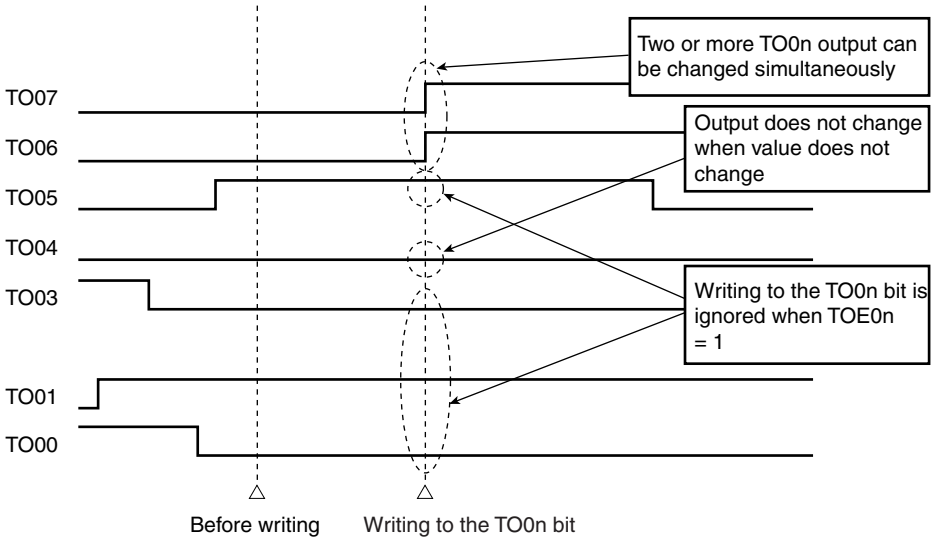
**Figure 6-36. Example of TO0n Bit Collective Manipulation**




Writing is done only to the TOmn bit with TOEmn = 0, and writing to the TOmn bit with TOEmn = 1 is ignored.

TOmn (channel output) to which TOEmn = 1 is set is not affected by the write operation. Even if the write operation is done to the TOmn bit, it is ignored and the output change by timer operation is normally done.

**Figure 6-37. TO0n Pin Statuses by Collective Manipulation of TO0n Bit**




(Caution and Remark are given on the next page.)

**Caution While timer output is enabled (TOEmn = 1), even if the output by timer interrupt of each timer (INTTMmn) contends with writing to the TOmn bit, output is normally done to the TOmn pin.**

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0, 1, 3 to 7)

### 6.6.5 Timer interrupt and TOmn pin output at operation start

In the interval timer mode or capture mode, the MDmn0 bit in timer mode register mn (TMRmn) sets whether or not to generate a timer interrupt at count start.

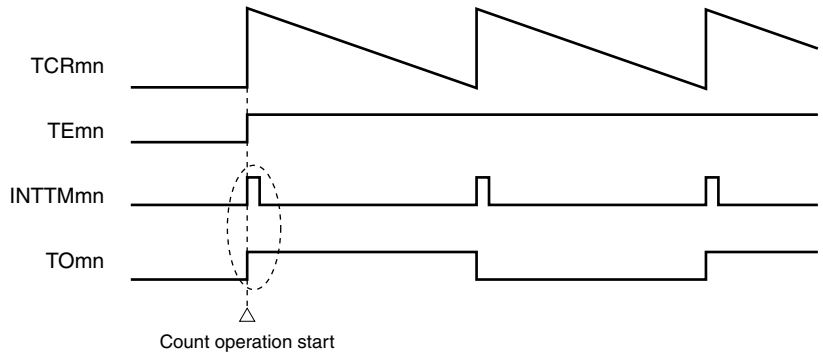
When MDmn0 is set to 1, the count operation start timing can be known by the timer interrupt (INTTMmn) generation.

In the other modes, neither timer interrupt at count operation start nor TOmn output is controlled.

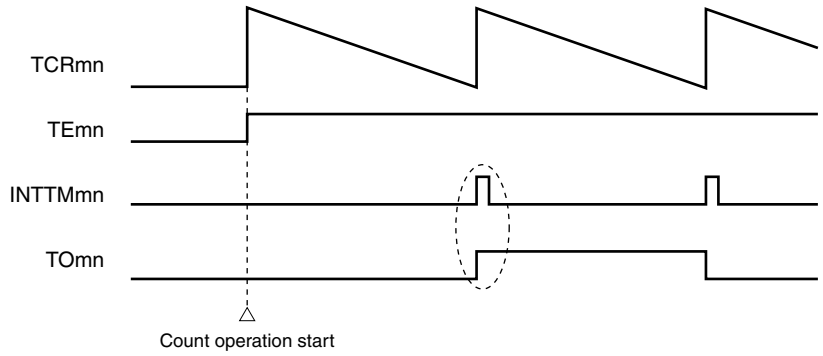
Figure 6-38 shows operation examples when the interval timer mode (TOEmn = 1, TOMmn = 0) is set.

**Figure 6-38. Operation Examples of Timer Interrupt at Count Operation Start and TOmn Output**

**(a) When MDmn0 is set to 1**




**(b) When MDmn0 is set to 0**




When MDmn0 is set to 1, a timer interrupt (INTTMmn) is output at count operation start, and TOmn performs a toggle operation.

When MDmn0 is set to 0, a timer interrupt (INTTMmn) is not output at count operation start, and TOmn does not change either. After counting one cycle, INTTMmn is output and TOmn performs a toggle operation.

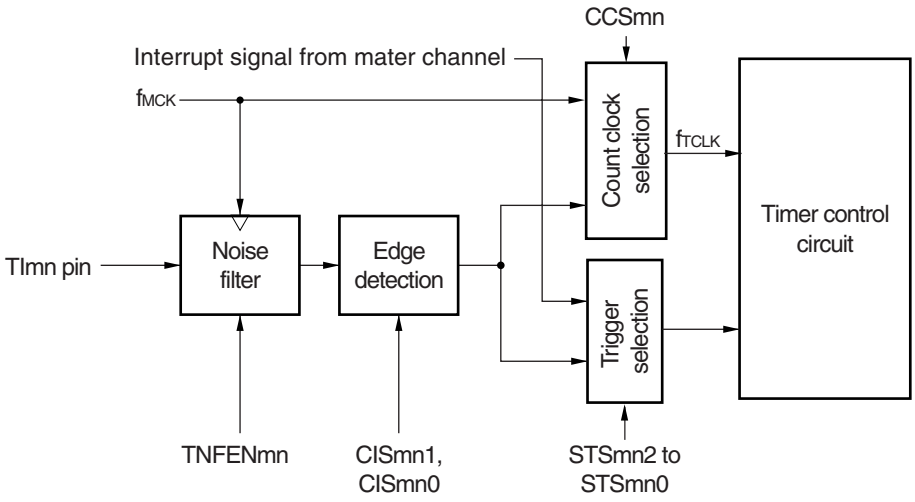
**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 1, 3 to 7)

## 6.7 Timer Input (TImn) Control

### 6.7.1 TImn pin input circuit configuration

The signal input from a timer input pin passes through a noise filter and edge detector, and is then input to the timer controller. If it is necessary to eliminate noise at the pin in question, enable its noise filter. The configuration of the input circuit is shown below.

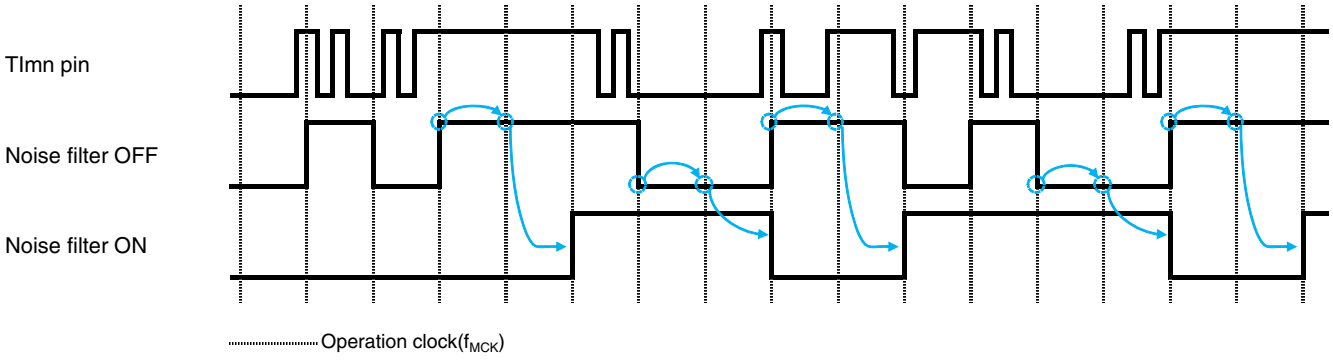
**Figure 6-39. Iutput Circuit Configuration**




### 6.7.2 Noise filter

If the noise filter is disabled, only synchronization is performed, based on the operating clock of channel n ($f_{MCK}$). If the noise filter is enabled, synchronization is performed based on the operating clock of channel n ($f_{MCK}$), and then two-clock match detection is performed. The following shows the waveforms of a signal after passing through a noise filter when the noise filter is enabled and disabled.

**Figure 6-40. Waveform of Sampling Signal Input to TImn Pin When Noise Filter Is Enabled and Disabled**



### 6.7.3 Cautions on channel input

When timer input pins are not used, the operating clock is not supplied to the noise filters. When use of a timer input pin is specified, therefore, the system must wait for the time shown below until the channel operation enable trigger flag for the channel corresponding to the timer input pin is set.

**(1) When the noise filter is disabled**

If bit 12 (CCSmn), bit 9 (STSmn1), or bit 8 (STSmn0) of timer mode register mn (TMRmn) is set when all of these bits are 0, wait for at least two cycles of the operating clock ($f_{MCK}$), and then set the corresponding timer operation enable trigger flag of the timer channel start register (TSm).

**(2) When the noise filter is enabled**

If bit 12 (CCSmn), bit 9 (STSmn1), or bit 8 (STSmn0) of timer mode register mn (TMRmn) is set when all of these bits are 0, wait for at least four cycles of the operating clock ($f_{MCK}$), and then set the corresponding timer operation enable trigger flag of the timer channel start register (TSm).

## 6.8 Independent Channel Operation Function of Timer Array Unit

### 6.8.1 Operation as interval timer/square wave output

**(1) Interval timer**

The timer array unit can be used as a reference timer that generates INTTMmn (timer interrupt) at fixed intervals. The interrupt generation period can be calculated by the following expression.

Generation period of INTTMmn (timer interrupt) = Period of count clock × (Set value of TDRmn + 1)

**(2) Operation as square wave output**

TOmn performs a toggle operation as soon as INTTMmn has been generated, and outputs a square wave with a duty factor of 50%.

The period and frequency for outputting a square wave from TOmn can be calculated by the following expressions.

- Period of square wave output from TOmn = Period of count clock × (Set value of TDRmn + 1) × 2
- Frequency of square wave output from TOmn = Frequency of count clock/{(Set value of TDRmn + 1) × 2}

Timer count register mn (TCRmn) operates as a down counter in the interval timer mode.

The TCRmn register loads the value of timer data register mn (TDRmn) at the first count clock after the channel start trigger bit (TSmn, TSHm1, TSHm3) of timer channel start register m (TSm) is set to 1. If the MDmn0 bit of timer mode register mn (TMRmn) is 0 at this time, INTTMmn is not output and TOmn is not toggled. If the MDmn0 bit of the TMRmn register is 1, INTTMmn is output and TOmn is toggled.

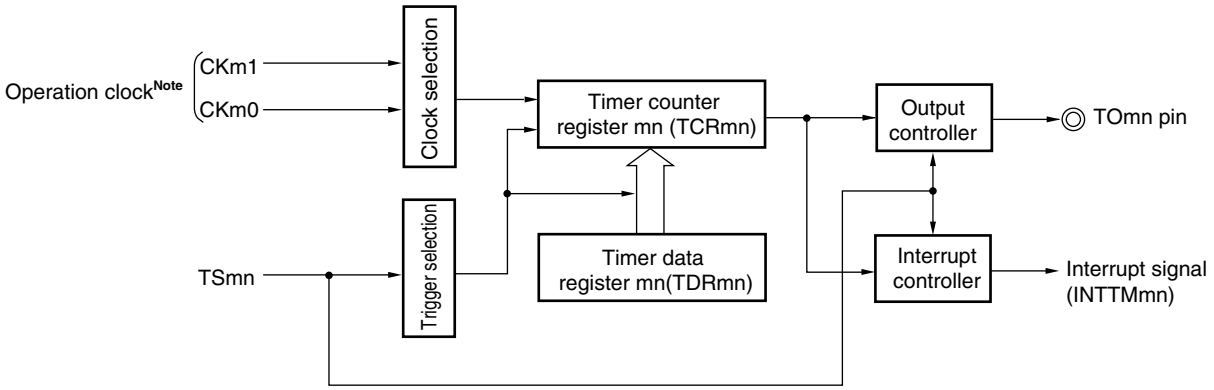
After that, the TCRmn register count down in synchronization with the count clock.

When TCRmn = 0000H, INTTMmn is output and TOmn is toggled at the next count clock. At the same time, the TCRmn register loads the value of the TDRmn register again. After that, the same operation is repeated.

The TDRmn register can be rewritten at any time. The new value of the TDRmn register becomes valid from the next period.

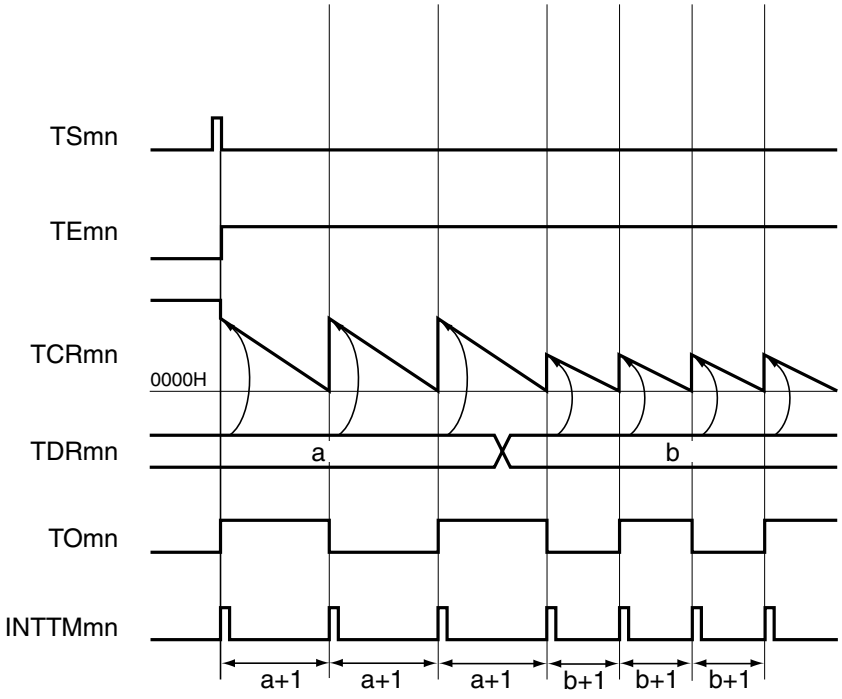
**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

**Figure 6-41. Block Diagram of Operation as Interval Timer/Square Wave Output**




**Note** When channels 1 and 3, the clock can be selected from CKm0, CKm1, CKm2 and CKm3.

**Figure 6-42. Example of Basic Timing of Operation as Interval Timer/Square Wave Output (MDmn0 = 1)**




**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

**2.** TSmn: Bit n of timer channel start register m (TSm)
TEmn: Bit n of timer channel enable status register m (TEm)
TCRmn: Timer count register mn (TCRmn)
TDRmn: Timer data register mn (TDRmn)
TOmn: TOmn pin output signal

**Figure 6-43. Example of Set Contents of Registers During Operation as Interval Timer/Square Wave Output (1/2)**

**(a) Timer mode register mn (TMRmn)**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn | CKSmn1 | CKSmn0 | | CCSmn | M/S[Note] | STSmn2 | STSmn1 | STSmn0 | CISmn1 | CISmn0 | | | MDmn3 | MDmn2 | MDmn1 | MDmn0 |
| | 1/0 | 1/0 | 0 | 0 | 0/1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1/0 |

Operation mode of channel n
000B: Interval timer

Setting of operation when counting is started
0: Neither generates INTTMmn nor inverts timer output when counting is started.
1: Generates INTTMmn and inverts timer output when counting is started.

Selection of TImn pin input edge
00B: Sets 00B because these are not used.

Start trigger selection
000B: Selects only software start.

Setting of MASTERmn bit (channels 2, 4, 6)
0: Independent channel operation function.
Setting of SPLITmn bit (channels 1, 3)
0: 16-bit timer mode
1: 8-bit timer mode

Count clock selection
0: Selects operation clock ($f_{MCK}$).

Operation clock ($f_{MCK}$) selection
00B: Selects CKm0 as operation clock of channel n.
10B: Selects CKm1 as operation clock of channel n.
01B: Selects CKm2 as operation clock of channels 1, 3 (This can only be selected channels 1 and 3).
11B: Selects CKm3 as operation clock of channels 1, 3 (This can only be selected channels 1 and 3).

**(b) Timer output register m (TOm)**

| | Bit n |
|---|---|
| TOm | TOmn |
| | 1/0 |

0: Outputs 0 from TOmn.
1: Outputs 1 from TOmn.

**(c) Timer output enable register m (TOEm)**

| | Bit n |
|---|---|
| TOEm | TOEmn |
| | 1/0 |

0: Stops the TOmn output operation by counting operation.
1: Enables the TOmn output operation by counting operation.

**Note** TMRm2, TMRm4, TMRm6: MASTERmn bit
TMRm1, TMRm3: SPLITmn bit
TMRm0, TMRm5, TMRm7: Fixed to 0

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

**Figure 6-43. Example of Set Contents of Registers During Operation as Interval Timer/Square Wave Output (2/2)**

**(d) Timer output level register m (TOLm)**

| | Bit n | |
|---|---|---|
| TOLm | TOLmn<br>0 | 0: Cleared to 0 when TOMmn = 0 (master channel output mode) |

**(e) Timer output mode register m (TOMm)**

| | Bit n | |
|---|---|---|
| TOMm | TOMmn<br>0 | 0: Sets master channel output mode. |

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

**Figure 6-44. Operation Procedure of Interval Timer/Square Wave Output Function (1/2)**

| | Software Operation | Hardware Status |
|---|---|---|
| TAU default setting | | Power-off status<br>(Clock supply is stopped and writing to each register is disabled.) |
| | Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1. → | Power-on status. Each channel stops operating.<br>(Clock supply is started and writing to each register is enabled.) |
| | Sets timer clock select register m (TPSm).<br>Determines clock frequencies of CKm0 to CKm3. | |
| Channel default setting | Sets timer mode register mn (TMRmn) (determines operation mode of channel).<br>Sets interval (period) value to timer data register mn (TDRmn). | Channel stops operating.<br>(Clock is supplied and some power is consumed.) |
| | To use the TOmn output<br>Clears the TOMmn bit of timer output mode register m (TOMm) to 0 (master channel output mode).<br>Clears the TOLmn bit to 0.<br>Sets the TOmn bit and determines default level of the TOmn output. → | The TOmn pin goes into Hi-Z output state.<br>The TOmn default setting level is output when the port mode register is in the output mode and the port register is 0. |
| | Sets the TOEmn bit to 1 and enables operation of TOmn. → | TOmn does not change because channel stops operating. |
| | Clears the port register and port mode register to 0. → | The TOmn pin outputs the TOmn set level. |
| Operation start | (Sets the TOEmn bit to 1 only if using TOmn output and resuming operation.).<br>Sets the TSmn (TSHm1, TSHm3) bit to 1. →<br>The TSmn (TSHm1, TSHm3) bit automatically returns to 0 because it is a trigger bit. | TEmn (TEHm1, TEHm3) = 1, and count operation starts.<br>Value of the TDRmn register is loaded to timer count register mn (TCRmn). INTTMmn is generated and TOmn performs toggle operation if the MDmn0 bit of the TMRmn register is 1. |
| During operation | Set value of the TDRmn register can be changed.<br>The TCRmn register can always be read.<br>The TSRmn register is not used.<br>Set values of the TOm and TOEm registers can be changed.<br>Set values of the TMRmn register, TOMmn, and TOLmn bits cannot be changed. | Counter (TCRmn) counts down. When count value reaches 0000H, the value of the TDRmn register is loaded to the TCRmn register again and the count operation is continued.<br>By detecting TCRmn = 0000H, INTTMmn is generated and TOmn performs toggle operation.<br>After that, the above operation is repeated. |
| Operation stop | The TTmn (TTHm1, TTHm3) bit is set to 1. →<br>The TTmn (TTHm1, TTHm3) bit automatically returns to 0 because it is a trigger bit. | TEmn (TEHm1, TEHm3), and count operation stops.<br>The TCRmn register holds count value and stops.<br>The TOmn output is not initialized but holds current status. |
| | The TOEmn bit is cleared to 0 and value is set to the TOmn bit. → | The TOmn pin outputs the TOmn bit set level. |

<R> Operation is resumed.

(**Remark** is listed on the next page.)

**Figure 6-44. Operation Procedure of Interval Timer/Square Wave Output Function (2/2)**

| | Software Operation | Hardware Status |
|---|---|---|
| TAU stop | To hold the TOmn pin output level<br>Clears the TOmn bit to 0 after the value to be held is set to the port register. → | The TOmn pin output level is held by port function. |
| | When holding the TOmn pin output level is not necessary<br>Setting not required. | |
| | The TAUmEN bit of the PER0 register is cleared to 0. → | Power-off status<br>All circuits are initialized and SFR of each channel is also initialized.<br>(The TOmn bit is cleared to 0 and the TOmn pin is set to port mode.) |

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

### 6.8.2 Operation as external event counter

The timer array unit can be used as an external event counter that counts the number of times the valid input edge (external event) is detected in the TImn pin. When a specified count value is reached, the event counter generates an interrupt. The specified number of counts can be calculated by the following expression.

| Specified number of counts = Set value of TDRmn + 1 |
|---|

Timer count register mn (TCRmn) operates as a down counter in the event counter mode.

The TCRmn register loads the value of timer data register mn (TDRmn) by setting any channel start trigger bit (TSmn, TSHm1, TSHm3) of timer channel start register m (TSm) to 1.

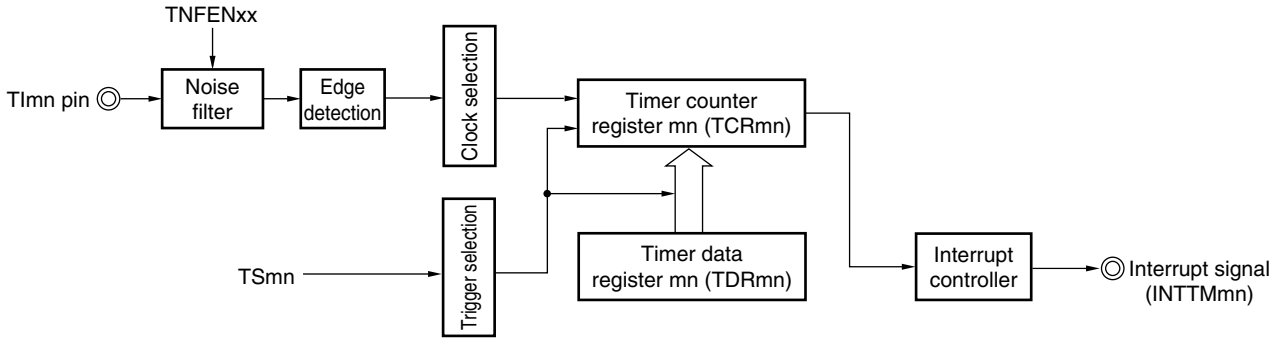
The TCRmn register counts down each time the valid input edge of the TImn pin has been detected. When TCRmn = 0000H, the TCRmn register loads the value of the TDRmn register again, and outputs INTTMmn.

After that, the above operation is repeated.

An irregular waveform that depends on external events is output from the TOmn pin. Stop the output by setting the TOEmn bit of timer output enable register m (TOEm) to 0.

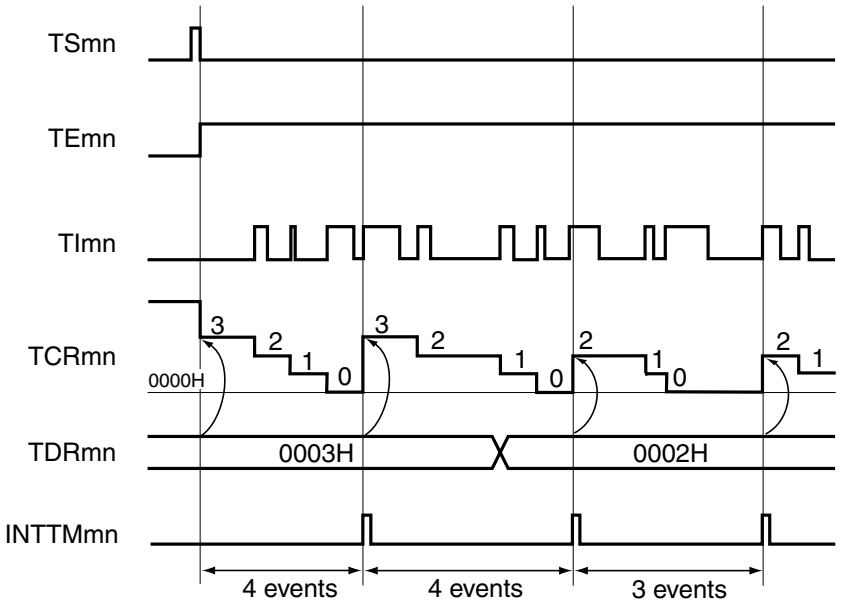
The TDRmn register can be rewritten at any time. The new value of the TDRmn register becomes valid during the next count period.

**Figure 6-45. Block Diagram of Operation as External Event Counter**




**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

**Figure 6-46. Example of Basic Timing of Operation as External Event Counter**

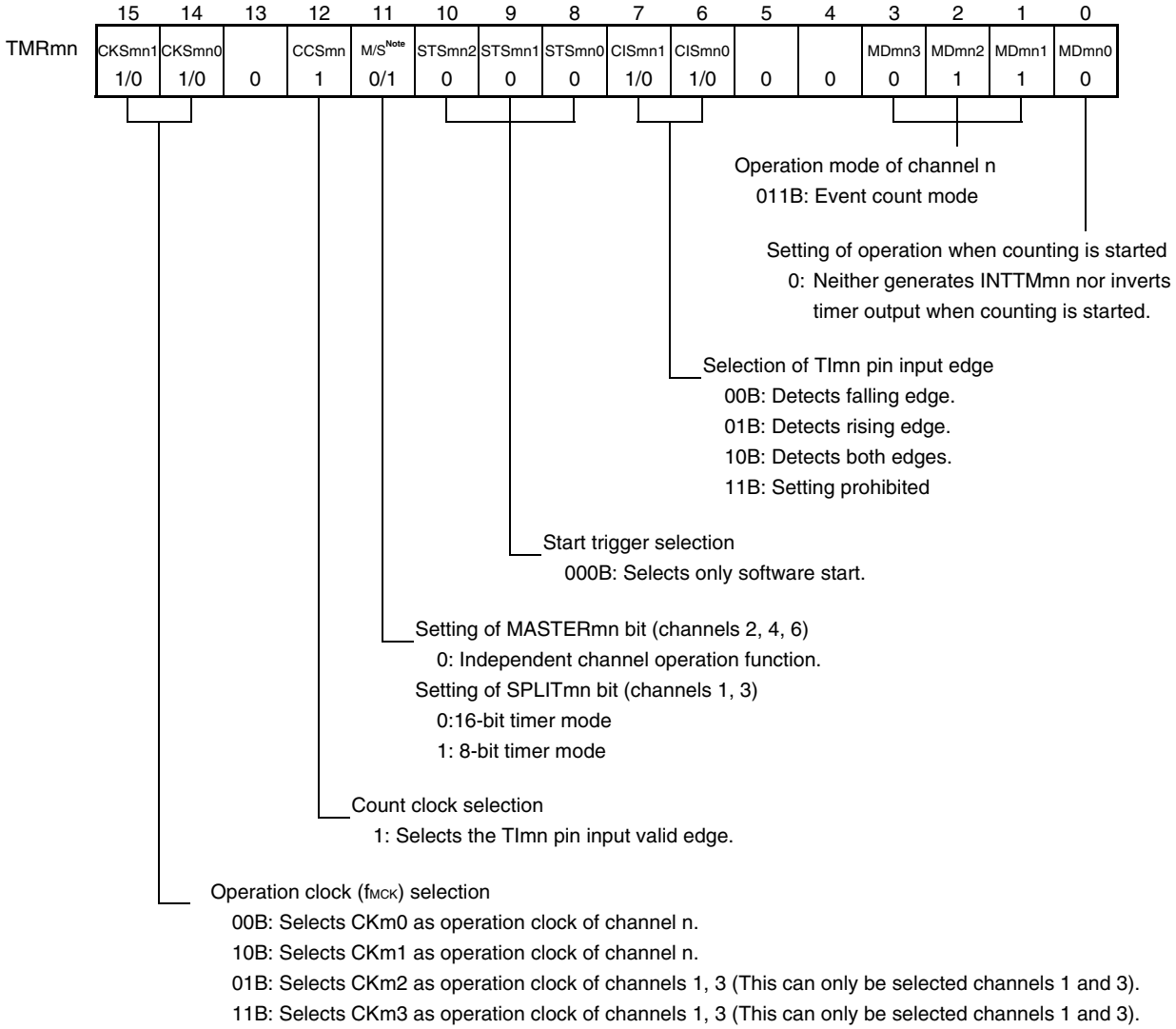



**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

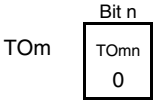
**2.** TSmn: Bit n of timer channel start register m (TSm)
TEmn: Bit n of timer channel enable status register m (TEm)
TImn: TImn pin input signal
TCRmn: Timer count register mn (TCRmn)
TDRmn: Timer data register mn (TDRmn)

**Figure 6-47. Example of Set Contents of Registers in External Event Counter Mode (1/2)**

**(a) Timer mode register mn (TMRmn)**

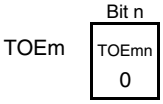



**(b) Timer output register m (TOm)**




0: Outputs 0 from TOmn.

**(c) Timer output enable register m (TOEm)**




0: Stops the TOmn output operation by counting operation.

**Note** TMRm2, TMRm4, TMRm6: MASTERmn bit
TMRm1, TMRm3: SPLITmn bit
TMRm0, TMRm5, TMRm7: Fixed to 0

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

**Figure 6-47. Example of Set Contents of Registers in External Event Counter Mode (2/2)**

**(d) Timer output level register m (TOLm)**

| | Bit n | |
|---|---|---|
| TOLm | TOLmn | 0: Cleared to 0 when TOMmn = 0 (master channel output mode). |
| | 0 | |

**(e) Timer output mode register m (TOMm)**

| | Bit n | |
|---|---|---|
| TOMm | TOMmn | 0: Sets master channel output mode. |
| | 0 | |

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

**Figure 6-48. Operation Procedure When External Event Counter Function Is Used**

| | Software Operation | Hardware Status |
|---|---|---|
| TAU default setting | | Power-off status<br>(Clock supply is stopped and writing to each register is disabled.) |
| | Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1. → | Power-on status. Each channel stops operating.<br>(Clock supply is started and writing to each register is enabled.) |
| | Sets timer clock select register m (TPSm).<br>Determines clock frequencies of CKm0 to CKm3. | |
| <R> Channel default setting | Sets corresponding bit of noise filter enable register 1 (NFEN1) to 0 (OFF) or 1 (ON).<br>Sets timer mode register mn (TMRmn) (determines operation mode of channel).<br>Sets number of counts to timer data register mn (TDRmn).<br>Clears the TOEmn bit of timer output enable register m (TOEm) to 0. | Channel stops operating.<br>(Clock is supplied and some power is consumed.) |
| Operation start | Sets the TSmn bit to 1. →<br>The TSmn bit automatically returns to 0 because it is a trigger bit. | TEmn = 1, and count operation starts.<br>Value of the TDRmn register is loaded to timer count register mn (TCRmn) and detection of the TImn pin input edge is awaited. |
| During operation | Set value of the TDRmn register can be changed.<br>The TCRmn register can always be read.<br>The TSRmn register is not used.<br>Set values of the TMRmn register, TOMmn, TOLmn, TOmn, and TOEmn bits cannot be changed. | Counter (TCRmn) counts down each time input edge of the TImn pin has been detected. When count value reaches 0000H, the value of the TDRmn register is loaded to the TCRmn register again, and the count operation is continued. By detecting TCRmn = 0000H, the INTTMmn output is generated.<br>After that, the above operation is repeated. |
| Operation stop | The TTmn bit is set to 1. →<br>The TTmn bit automatically returns to 0 because it is a trigger bit. | TEmn = 0, and count operation stops.<br>The TCRmn register holds count value and stops. |
| TAU stop | The TAUmEN bit of the PER0 register is cleared to 0. → | Power-off status<br>All circuits are initialized and SFR of each channel is also initialized. |

Operation is resumed.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

### 6.8.3 Operation as frequency divider (channel 0 of unit 0 only)

The timer array unit can be used as a frequency divider that divides a clock input to the TI00 pin and outputs the result from the TO00 pin.

The divided clock frequency output from TO00 can be calculated by the following expression.

- When rising edge/falling edge is selected:
  Divided clock frequency = Input clock frequency/{(Set value of TDR00 + 1) × 2}
- When both edges are selected:
  Divided clock frequency ≅ Input clock frequency/(Set value of TDR00 + 1)

Timer count register 00 (TCR00) operates as a down counter in the interval timer mode.

After the channel start trigger bit (TS00) of timer channel start register 0 (TS0) is set to 1, the TCR00 register loads the value of timer data register 00 (TDR00) when the TI00 valid edge is detected.

If the MD000 bit of timer mode register 00 (TMR00) is 0 at this time, INTTM00 is not output and TO00 is not toggled. If the MD000 bit of timer mode register 00 (TMR00) is 1, INTTM00 is output and TO00 is toggled.

After that, the TCR00 register counts down at the valid edge of the TI00 pin. When TCR00 = 0000H, it toggles TO00. At the same time, the TCR00 register loads the value of the TDR00 register again, and continues counting.

If detection of both the edges of the TI00 pin is selected, the duty factor error of the input clock affects the divided clock period of the TO00 output.

The period of the TO00 output clock includes a sampling error of one period of the operation clock.

Clock period of TO00 output = Ideal TO00 output clock period ± Operation clock period (error)

The TDR00 register can be rewritten at any time. The new value of the TDR00 register becomes valid during the next count period.

**Figure 6-49. Block Diagram of Operation as Frequency Divider**

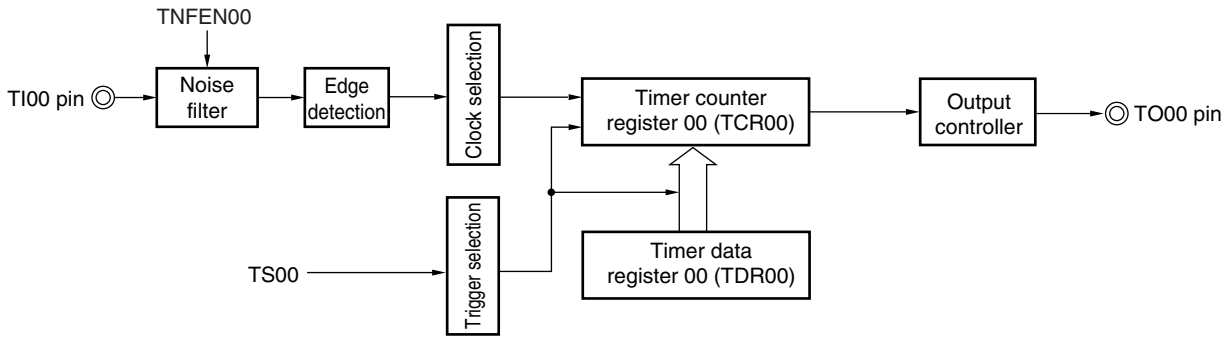
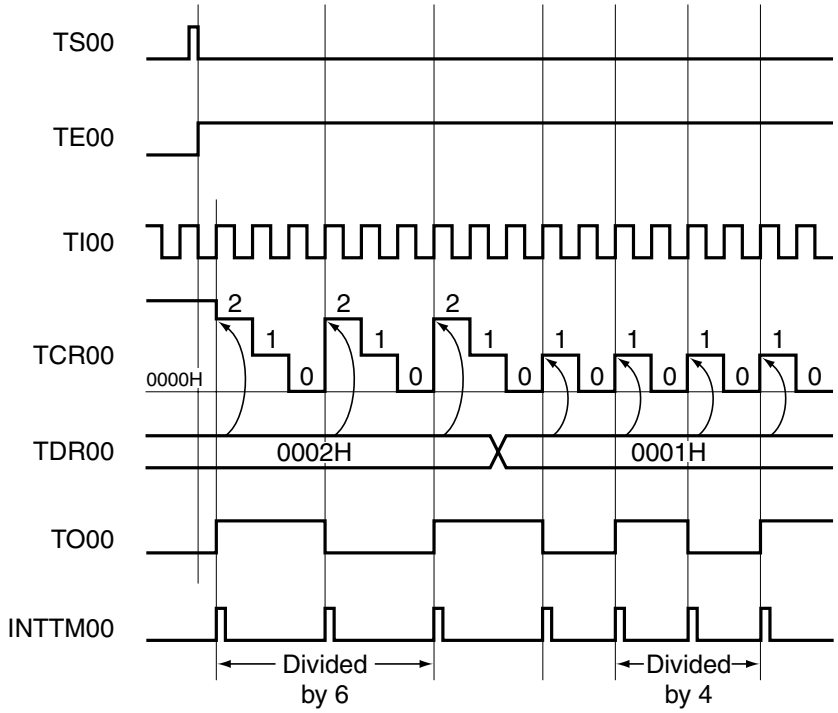


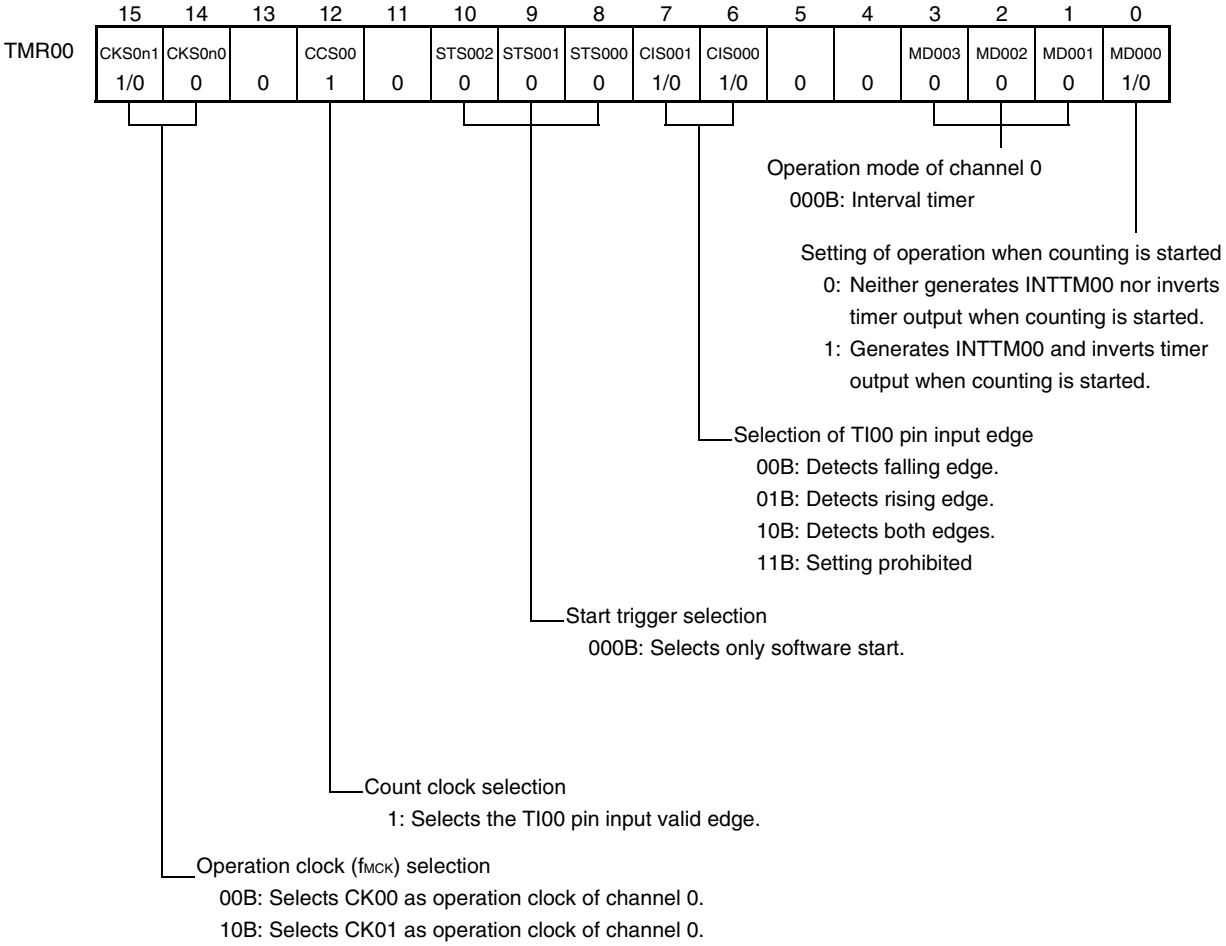
**Figure 6-50. Example of Basic Timing of Operation as Frequency Divider (MD000 = 1)**




**Remark** TS00: Bit n of timer channel start register 0 (TS0)
TE00: Bit n of timer channel enable status register 0 (TE0)
TI00: TI00 pin input signal
TCR00: Timer count register 00 (TCR00)
TDR00: Timer data register 00 (TDR00)
TO00: TO00 pin output signal

**Figure 6-51. Example of Set Contents of Registers During Operation as Frequency Divider**

<R> **(a) Timer mode register 00 (TMR00)**




**(b) Timer output register 0 (TO0)**

| | Bit 0 | |
|---|---|---|
| TO0 | TO00 | 0: Outputs 0 from TO00.<br>1: Outputs 1 from TO00. |
| | 1/0 | |

**(c) Timer output enable register 0 (TOE0)**

| | Bit 0 | |
|---|---|---|
| TOE0 | TOE00 | 0: Stops the TO00 output operation by counting operation.<br>1: Enables the TO00 output operation by counting operation. |
| | 1/0 | |

**(d) Timer output level register 0 (TOL0)**

| | Bit 0 | |
|---|---|---|
| TOL0 | TOL00 | 0: Cleared to 0 when TOM00 = 0 (master channel output mode) |
| | 0 | |

**(e) Timer output mode register 0 (TOM0)**

| | Bit 0 | |
|---|---|---|
| TOM0 | TOM00 | 0: Sets master channel output mode. |
| | 0 | |

**Figure 6-52. Operation Procedure When Frequency Divider Function Is Used**

| | Software Operation | Hardware Status |
|---|---|---|
| TAU default setting | | Power-off status<br>(Clock supply is stopped and writing to each register is disabled.) |
| | Sets the TAU0EN bit of peripheral enable register 0 (PER0) to 1. → | Power-on status. Each channel stops operating.<br>(Clock supply is started and writing to each register is enabled.) |
| | Sets timer clock select register 0 (TPS0).<br>Determines clock frequencies of CK00 to CK03. | |
| <R> Channel default setting | Sets corresponding bit of noise filter enable register 1 (NFEN1) to 0 (OFF) or 1 (ON).<br>Sets timer mode register 0n (TMR0n) (determines operation mode of channel and selects the detection edge).<br>Sets interval (period) value to timer data register 00 (TDR00). | Channel stops operating.<br>(Clock is supplied and some power is consumed.) |
| | Clears the TOM00 bit of timer output mode register 0 (TOM0) to 0 (master channel output mode).<br>Clears the TOL00 bit to 0.<br>Sets the TO00 bit and determines default level of the TO00 output. → | The TO00 pin goes into Hi-Z output state.<br>The TO00 default setting level is output when the port mode register is in output mode and the port register is 0. |
| | Sets the TOE00 bit to 1 and enables operation of TO00. → | TO00 does not change because channel stops operating. |
| | Clears the port register and port mode register to 0. → | The TO00 pin outputs the TO00 set level. |
| Operation start | Sets the TOE00 bit to 1 (only when operation is resumed). →<br>Sets the TS00 bit to 1.<br>The TS00 bit automatically returns to 0 because it is a trigger bit. | TE00 = 1, and count operation starts.<br>Value of the TDR00 register is loaded to timer count register 00 (TCR00). INTTM00 is generated and TO00 performs toggle operation if the MD000 bit of the TMR00 register is 1. |
| During operation | Set value of the TDR00 register can be changed.<br>The TCR00 register can always be read.<br>The TSR00 register is not used.<br>Set values of the TO0 and TOE0 registers can be changed.<br>Set values of the TMR00 register, TOM00, and TOL00 bits cannot be changed. | Counter (TCR00) counts down. When count value reaches 0000H, the value of the TDR00 register is loaded to the TCR00 register again, and the count operation is continued.<br>By detecting TCR00 = 0000H, INTTM00 is generated and TO00 performs toggle operation.<br>After that, the above operation is repeated. |
| Operation stop | The TT00 bit is set to 1. →<br>The TT00 bit automatically returns to 0 because it is a trigger bit. | TE00 = 0, and count operation stops.<br>The TCR00 register holds count value and stops.<br>The TO00 output is not initialized but holds current status. |
| | The TOE00 bit is cleared to 0 and value is set to the TO00 bit. → | The TO00 pin outputs the TO00 set level. |
| TAU stop | To hold the TO00 pin output level<br>Clears the TO00 bit to 0 after the value to be held is set to the port register. →<br>When holding the TO00 pin output level is not necessary<br>Setting not required. | The TO00 pin output level is held by port function. |
| | The TAU0EN bit of the PER0 register is cleared to 0. → | Power-off status<br>All circuits are initialized and SFR of each channel is also initialized.<br>(The TO00 bit is cleared to 0 and the TO00 pin is set to port mode). |

Operation is resumed.

### 6.8.4 Operation as input pulse interval measurement

The count value can be captured at the TImn valid edge and the interval of the pulse input to TImn can be measured.

<R> In addition, the count value can be captured by using software operation (TSmn = 1) as a capture trigger while the TEmn bit is set to 1.

The pulse interval can be calculated by the following expression.

| TImn input pulse interval = Period of count clock × ((10000H × TSRmn: OVF) + (Capture value of TDRmn + 1)) |
|---|

**Caution The TImn pin input is sampled using the operating clock selected with the CKSmn bit of timer mode register mn (TMRmn), so an error of up to one operating clock cycle occurs.**

Timer count register mn (TCRmn) operates as an up counter in the capture mode.

When the channel start trigger bit (TSmn) of timer channel start register m (TSm) is set to 1, the TCRmn register counts up from 0000H in synchronization with the count clock.

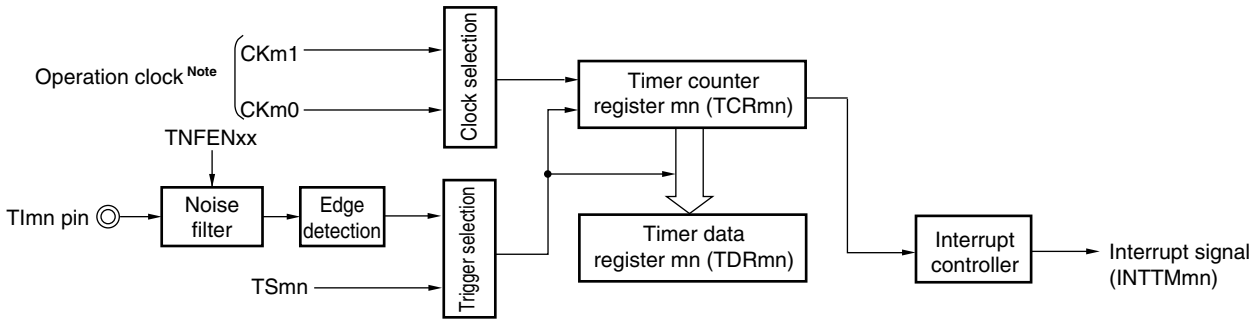
When the TImn pin input valid edge is detected, the count value of the TCRmn register is transferred (captured) to timer data register mn (TDRmn) and, at the same time, the TCRmn register is cleared to 0000H, and the INTTMmn is output. If the counter overflows at this time, the OVF bit of timer status register mn (TSRmn) is set to 1. If the counter does not overflow, the OVF bit is cleared. After that, the above operation is repeated.

As soon as the count value has been captured to the TDRmn register, the OVF bit of the TSRmn register is updated depending on whether the counter overflows during the measurement period. Therefore, the overflow status of the captured value can be checked.

If the counter reaches a full count for two or more periods, it is judged to be an overflow occurrence, and the OVF bit of the TSRmn register is set to 1. However, a normal interval value cannot be measured for the OVF bit, if two or more overflows occur.

Set the STSmn2 to STSmn0 bits of the TMRmn register to 001B to use the valid edges of TImn as a start trigger and a capture trigger.

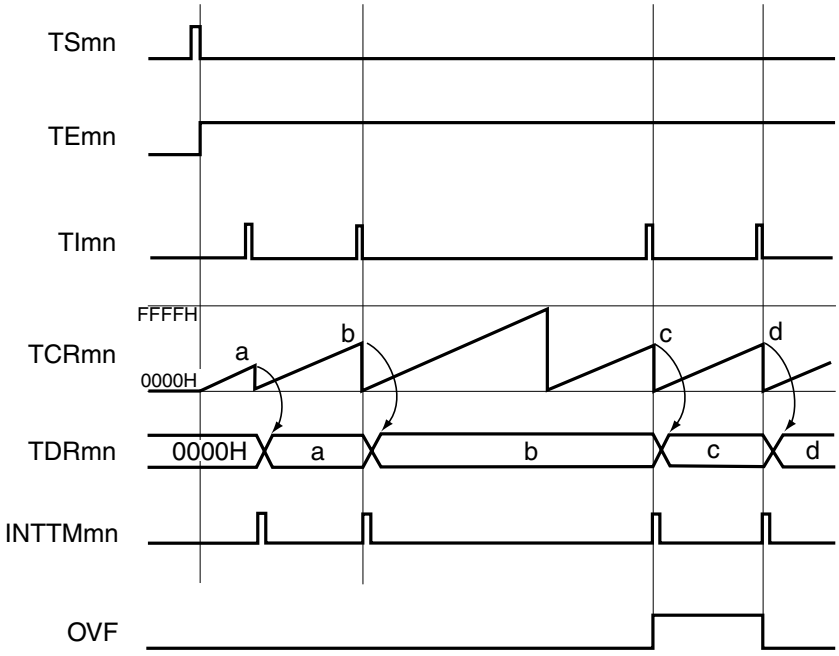
**Figure 6-53. Block Diagram of Operation as Input Pulse Interval Measurement**




**Note** When channels 1 and 3, the clock can be selected from CKm0, CKm1, CKm2 and CKm3.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

**Figure 6-54. Example of Basic Timing of Operation as Input Pulse Interval Measurement (MDmn0 = 0)**

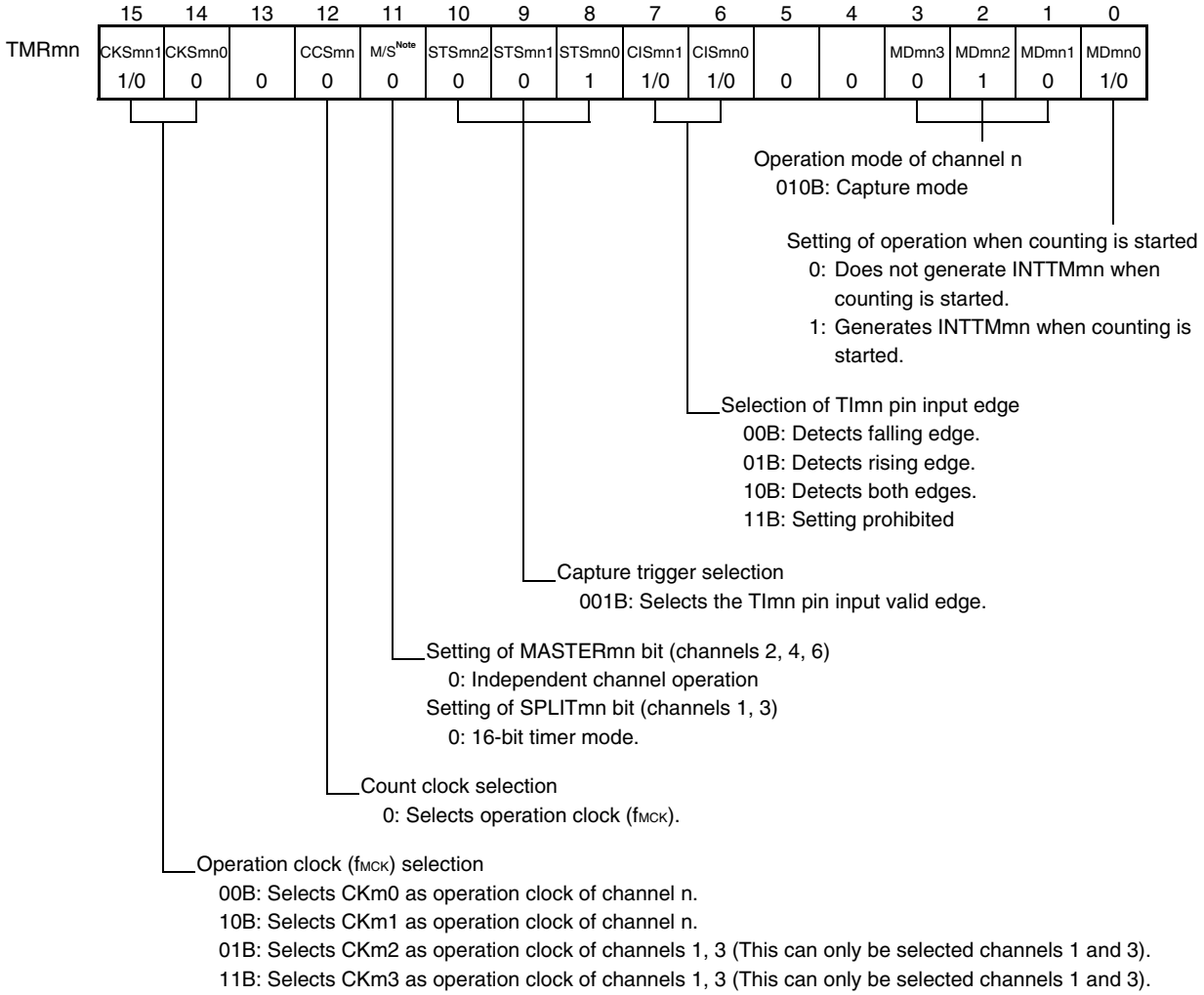



**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

**2.** TSmn: Bit n of timer channel start register m (TSm)
TEmn: Bit n of timer channel enable status register m (TEm)
TImn: TImn pin input signal
TCRmn: Timer count register mn (TCRmn)
TDRmn: Timer data register mn (TDRmn)
OVF: Bit 0 of timer status register mn (TSRmn)

**Figure 6-55. Example of Set Contents of Registers to Measure Input Pulse Interval**

**(a) Timer mode register mn (TMRmn)**




**(b) Timer output register m (TOm)**

| | Bit n | |
|---|---|---|
| TOm | TOmn<br>0 | 0: Outputs 0 from TOmn. |

**(c) Timer output enable register m (TOEm)**

| | Bit n | |
|---|---|---|
| TOEm | TOEmn<br>0 | 0: Stops TOmn output operation by counting operation. |

**(d) Timer output level register m (TOLm)**

| | Bit n | |
|---|---|---|
| TOLm | TOLmn<br>0 | 0: Cleared to 0 when TOMmn = 0 (master channel output mode). |

**(e) Timer output mode register m (TOMm)**

| | Bit n | |
|---|---|---|
| TOMm | TOMmn<br>0 | 0: Sets master channel output mode. |

**Note** TMRm2, TMRm4, TMRm6: MASTERmn bit
TMRm1, TMRm3: SPLITmn bit
TMRm0, TMRm5, TMRm7: Fixed to 0

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

**Figure 6-56. Operation Procedure When Input Pulse Interval Measurement Function Is Used**

| | Software Operation | Hardware Status |
|---|---|---|
| TAU default setting | | Power-off status<br>(Clock supply is stopped and writing to each register is disabled.) |
| | Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1. → | Power-on status. Each channel stops operating.<br>(Clock supply is started and writing to each register is enabled.) |
| | Sets timer clock select register m (TPSm).<br>Determines clock frequencies of CKm0 to CKm3. | |
| <R> Channel default setting | Sets corresponding bit of noise filter enable register 1 (NFEN1) to 0 (OFF) or 1 (ON).<br>Sets timer mode register mn (TMRmn) (determines operation mode of channel). | Channel stops operating.<br>(Clock is supplied and some power is consumed.) |
| Operation start | Sets TSmn bit to 1. →<br>The TSmn bit automatically returns to 0 because it is a trigger bit. | TEmn = 1, and count operation starts.<br>Timer count register mn (TCRmn) is cleared to 0000H.<br>When the MDmn0 bit of the TMRmn register is 1, INTTMmn is generated. |
| <R> During operation | Set values of only the CISmn1 and CISmn0 bits of the TMRmn register can be changed.<br>The TDRmn register can always be read.<br>The TCRmn register can always be read.<br>The TSRmn register can always be read.<br>Set values of the TOMmn, TOLmn, TOmn, and TOEmn bits cannot be changed. | Counter (TCRmn) counts up from 0000H. When the valid edge of the TImn pin input is detected or the TSmn bit is set to 1, the count value is transferred (captured) to timer data register mn (TDRmn). At the same time, the TCRmn register is cleared to 0000H, and the INTTMmn signal is generated.<br>If an overflow occurs at this time, the OVF bit of timer status register mn (TSRmn) is set; if an overflow does not occur, the OVF bit is cleared.<br>After that, the above operation is repeated. |
| Operation stop | The TTmn bit is set to 1. →<br>The TTmn bit automatically returns to 0 because it is a trigger bit. | TEmn = 0, and count operation stops.<br>The TCRmn register holds count value and stops.<br>The OVF bit of the TSRmn register is also held. |
| TAU stop | The TAUmEN bit of the PER0 register is cleared to 0. → | Power-off status<br>All circuits are initialized and SFR of each channel is also initialized. |

<R> Operation is resumed.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

### 6.8.5 Operation as input signal high-/low-level width measurement

**Caution When using a channel to implement the LIN-bus, set bit 1 (ISC1) of the input switch control register (ISC) to 1. In the following descriptions, read TImn as RxD2.**

By starting counting at one edge of the TImn pin input and capturing the number of counts at another edge, the signal width (high-level width/low-level width) of TImn can be measured. The signal width of TImn can be calculated by the following expression.

| Signal width of TImn input = Period of count clock × ((10000H × TSRmn: OVF) + (Capture value of TDRmn + 1)) |
|---|

**Caution The TImn pin input is sampled using the operating clock selected with the CKSmn bit of timer mode register mn (TMRmn), so an error equivalent to one operation clock occurs.**

Timer count register mn (TCRmn) operates as an up counter in the capture & one-count mode.

When the channel start trigger bit (TSmn) of timer channel start register m (TSm) is set to 1, the TEmn bit is set to 1 and the TImn pin start edge detection wait status is set.

When the TImn pin input start edge (rising edge of the TImn pin input when the high-level width is to be measured) is detected, the counter counts up from 0000H in synchronization with the count clock. When the valid capture edge (falling edge of the TImn pin input when the high-level width is to be measured) is detected later, the count value is transferred to timer data register mn (TDRmn) and, at the same time, INTTMmn is output. If the counter overflows at this time, the OVF bit of timer status register mn (TSRmn) is set to 1. If the counter does not overflow, the OVF bit is cleared. The TCRmn register stops at the value “value transferred to the TDRmn register + 1”, and the TImn pin start edge detection wait status is set. After that, the above operation is repeated.

As soon as the count value has been captured to the TDRmn register, the OVF bit of the TSRmn register is updated depending on whether the counter overflows during the measurement period. Therefore, the overflow status of the captured value can be checked.

If the counter reaches a full count for two or more periods, it is judged to be an overflow occurrence, and the OVF bit of the TSRmn register is set to 1. However, a normal interval value cannot be measured for the OVF bit, if two or more overflows occur.

Whether the high-level width or low-level width of the TImn pin is to be measured can be selected by using the CISmn1 and CISmn0 bits of the TMRmn register.

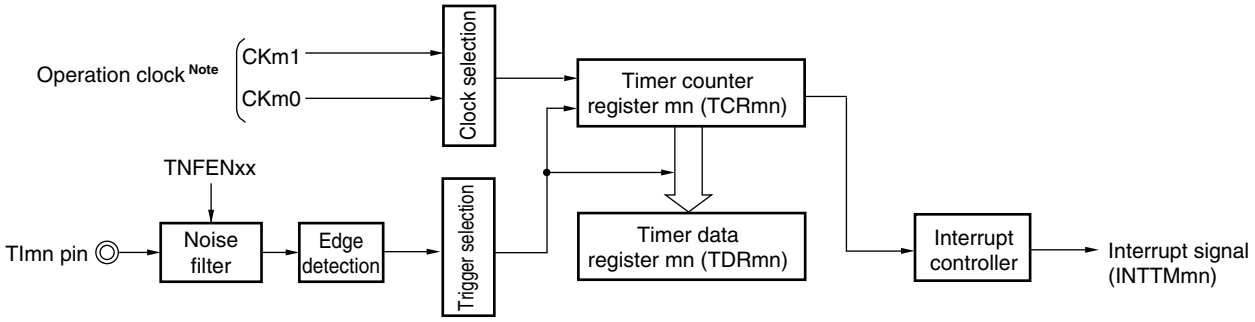
Because this function is used to measure the signal width of the TImn pin input, the TSmn bit cannot be set to 1 while the TEmn bit is 1.

CISmn1, CISmn0 of TMRmn register = 10B: Low-level width is measured.
CISmn1, CISmn0 of TMRmn register = 11B: High-level width is measured.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

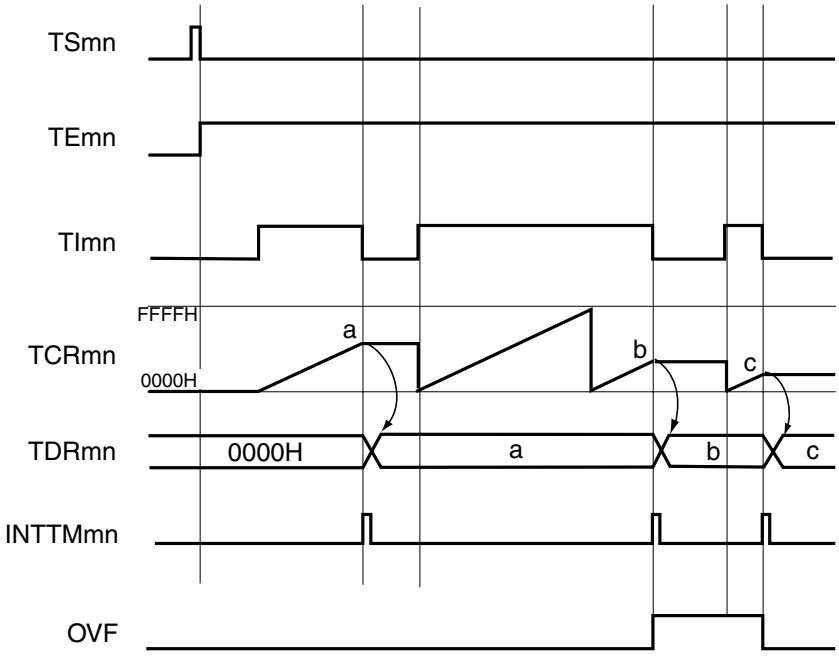
**Figure 6-57. Block Diagram of Operation as Input Signal High-/Low-Level Width Measurement**

<R>




**Note** For channels 1 and 3, the clock can be selected from CKm0, CKm1, CKm2 and CKm3.

**Figure 6-58. Example of Basic Timing of Operation as Input Signal High-/Low-Level Width Measurement**

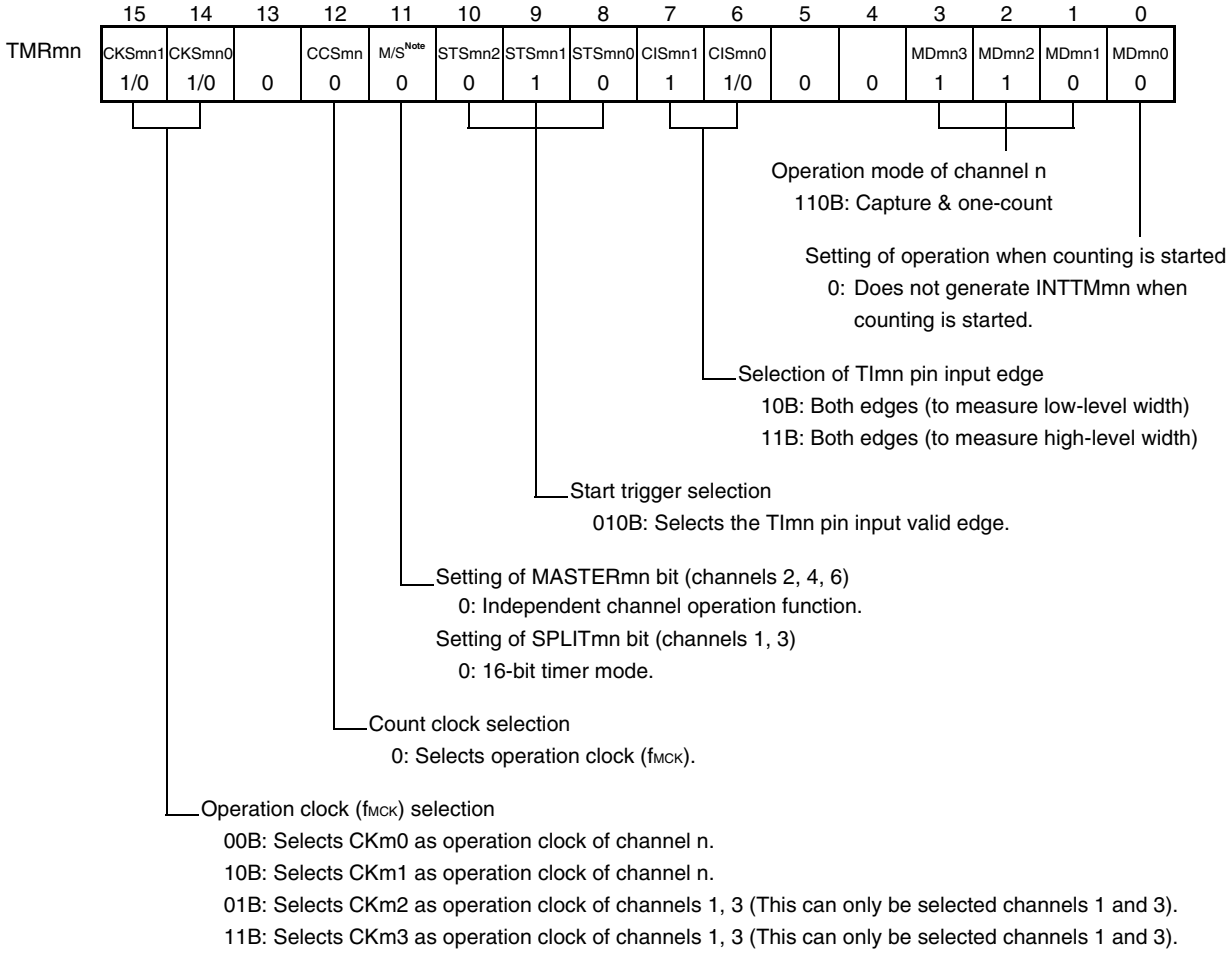



**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

**2.** TSmn: Bit n of timer channel start register m (TSm)
TEmn: Bit n of timer channel enable status register m (TEm)
TImn: TImn pin input signal
TCRmn: Timer count register mn (TCRmn)
TDRmn: Timer data register mn (TDRmn)
OVF: Bit 0 of timer status register mn (TSRmn)

**Figure 6-59. Example of Set Contents of Registers to Measure Input Signal High-/Low-Level Width**

**(a) Timer mode register mn (TMRmn)**




**(b) Timer output register m (TOm)**

| | Bit n | |
|---|---|---|
| TOm | TOmn<br>0 | 0: Outputs 0 from TOmn. |

**(c) Timer output enable register m (TOEm)**

| | Bit n | |
|---|---|---|
| TOEm | TOEmn<br>0 | 0: Stops the TOmn output operation by counting operation. |

**(d) Timer output level register m (TOLm)**

| | Bit n | |
|---|---|---|
| TOLm | TOLmn<br>0 | 0: Cleared to 0 when TOMmn = 0 (master channel output mode). |

**(e) Timer output mode register m (TOMm)**

| | Bit n | |
|---|---|---|
| TOMm | TOMmn<br>0 | 0: Sets master channel output mode. |

**Note** TMRm2, TMRm4, TMRm6: MASTERmn bit
TMRm1, TMRm3: SPLITmn bit
TMRm0, TMRm5, TMRm7: Fixed to 0

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

**Figure 6-60. Operation Procedure When Input Signal High-/Low-Level Width Measurement Function Is Used**

| | Software Operation | Hardware Status |
|---|---|---|
| TAU default setting | | Power-off status<br>(Clock supply is stopped and writing to each register is disabled.) |
| | Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1. → | Power-on status. Each channel stops operating.<br>(Clock supply is started and writing to each register is enabled.) |
| | Sets timer clock select register m (TPSm).<br>Determines clock frequencies of CKm0 to CKm3. | |
| <R> Channel default setting | Sets corresponding bit of noise filter enable register 1 (NFEN1) to 0 (OFF) or 1 (ON).<br>Sets timer mode register mn (TMRmn) (determines operation mode of channel).<br>Clears the TOEmn bit to 0 and stops operation of TOmn. | Channel stops operating.<br>(Clock is supplied and some power is consumed.) |
| Operation start | Sets the TSmn bit to 1. →<br>The TSmn bit automatically returns to 0 because it is a trigger bit. | TEmn = 1, and the TImn pin start edge detection wait status is set. |
| | Detects the TImn pin input count start valid edge. → | Clears timer count register mn (TCRmn) to 0000H and starts counting up. |
| During operation | Set value of the TDRmn register can be changed.<br>The TCRmn register can always be read.<br>The TSRmn register is not used.<br>Set values of the TMRmn register, TOMmn, TOLmn, TOmn, and TOEmn bits cannot be changed. | When the TImn pin start edge is detected, the counter (TCRmn) counts up from 0000H. If a capture edge of the TImn pin is detected, the count value is transferred to timer data register mn (TDRmn) and INTTMmn is generated.<br>If an overflow occurs at this time, the OVF bit of timer status register mn (TSRmn) is set; if an overflow does not occur, the OVF bit is cleared. The TCRmn register stops the count operation until the next TImn pin start edge is detected. |
| Operation stop | The TTmn bit is set to 1. →<br>The TTmn bit automatically returns to 0 because it is a trigger bit. | TEmn = 0, and count operation stops.<br>The TCRmn register holds count value and stops.<br>The OVF bit of the TSRmn register is also held. |
| TAU stop | The TAUmEN bit of the PER0 register is cleared to 0. → | Power-off status<br>All circuits are initialized and SFR of each channel is also initialized. |

Operation is resumed.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

### 6.8.6 Operation as delay counter

It is possible to start counting down when the valid edge of the TImn pin input is detected (an external event), and then generate INTTMmn (a timer interrupt) after any specified interval.

It is also possible to start counting down and generate INTTMmn (timer interrupt) at any interval by setting TSmn to 1 by software while TEmn = 1.

The interrupt generation period can be calculated by the following expression.

| Generation period of INTTMmn (timer interrupt) = Period of count clock × (Set value of TDRmn + 1) |
|---|

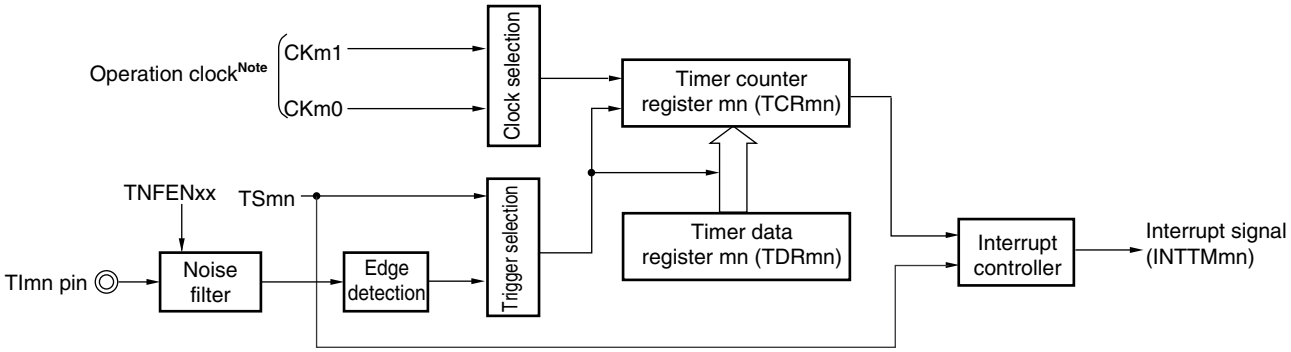
Timer count register mn (TCRmn) operates as a down counter in the one-count mode.

When the channel start trigger bit (TSmn, TSHm1, TSHm3) of timer channel start register m (TSm) is set to 1, the TEmn, TEHm1, TEHm3 bits are set to 1 and the TImn pin input valid edge detection wait status is set.

Timer count register mn (TCRmn) starts operating upon TImn pin input valid edge detection and loads the value of timer data register mn (TDRmn). The TCRmn register counts down from the value of the TDRmn register it has loaded, in synchronization with the count clock. When TCRmn = 0000H, it outputs INTTMmn and stops counting until the next TImn pin input valid edge is detected.

The TDRmn register can be rewritten at any time. The new value of the TDRmn register becomes valid from the next period.

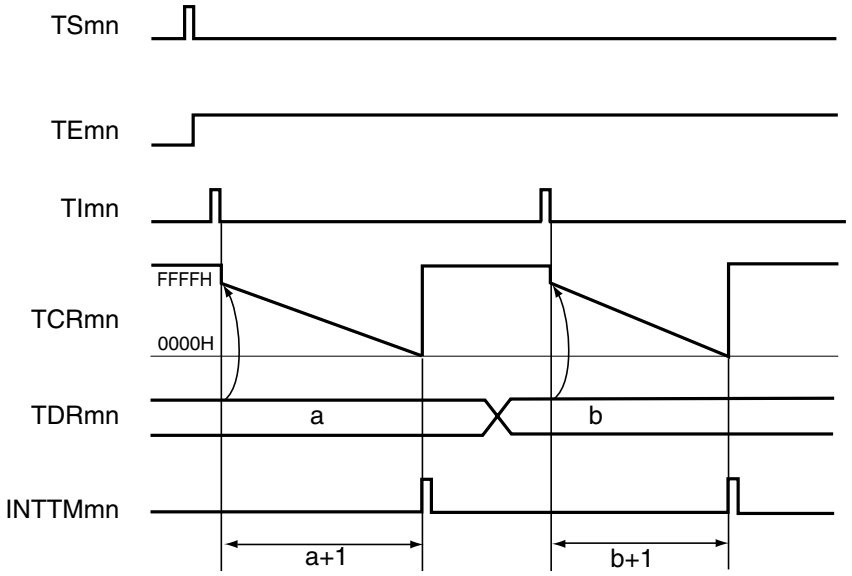
**Figure 6-61. Block Diagram of Operation as Delay Counter**




**Note** For using channels 1 and 3, the clock can be selected from CKm0, CKm1, CKm2 and CKm3.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

**Figure 6-62. Example of Basic Timing of Operation as Delay Counter**

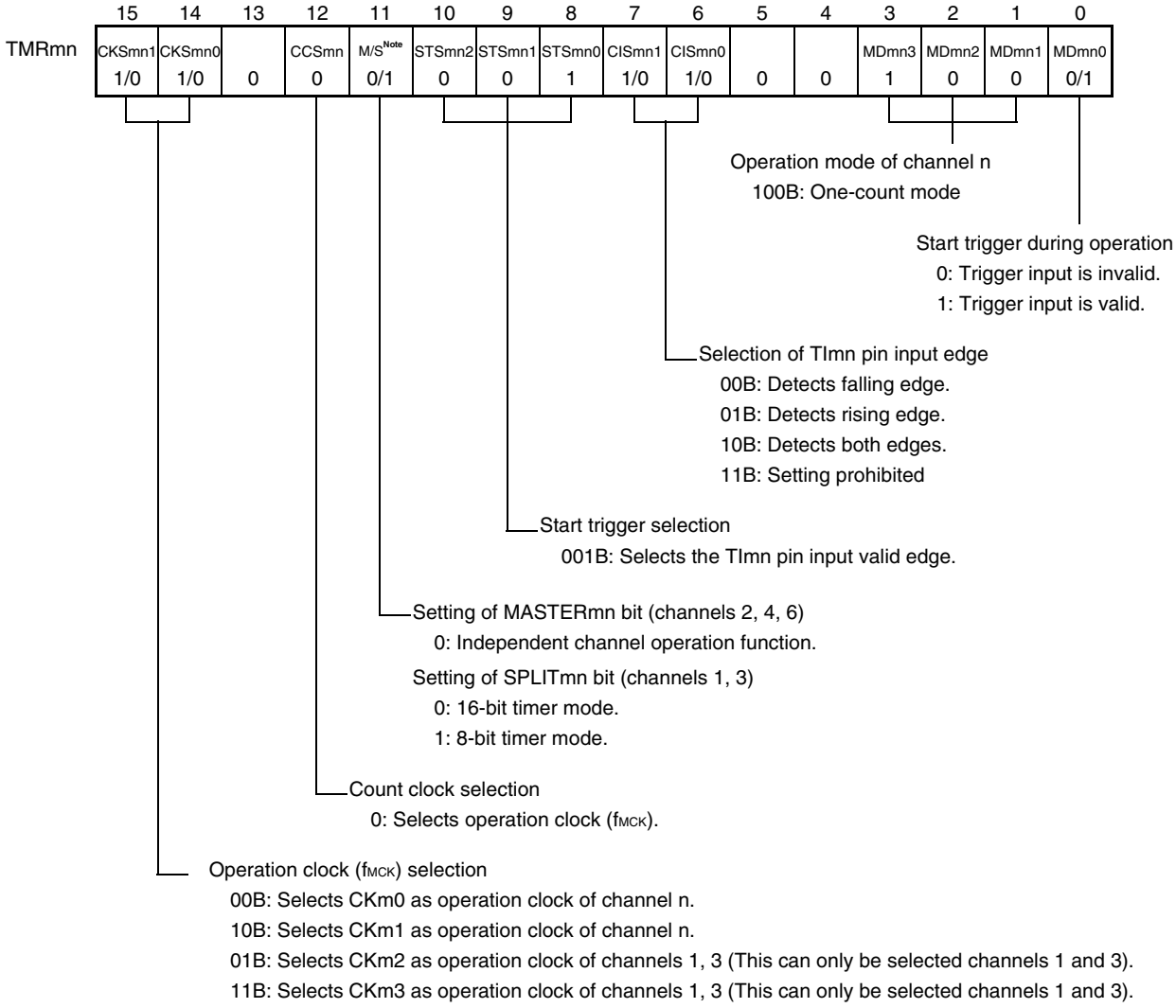



**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

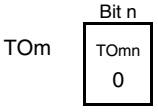
**2.** TSmn: Bit n of timer channel start register m (TSm)
TEmn: Bit n of timer channel enable status register m (TEm)
TImn: TImn pin input signal
TCRmn: Timer count register mn (TCRmn)
TDRmn: Timer data register mn (TDRmn)

**Figure 6-63. Example of Set Contents of Registers to Delay Counter (1/2)**

**(a) Timer mode register mn (TMRmn)**

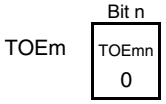



**(b) Timer output register m (TOm)**




0: Outputs 0 from TOmn.

**(c) Timer output enable register m (TOEm)**




0: Stops the TOmn output operation by counting operation.

**Note** TMRm2, TMRm4, TMRm6: MASTERmn bit
TMRm1, TMRm3: SPLITmn bit
TMRm0, TMRm5, TMRm7: Fixed to 0

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

**Figure 6-63. Example of Set Contents of Registers to Delay Counter (2/2)**

**(d) Timer output level register m (TOLm)**

| | Bit n | |
|---|---|---|
| TOLm | TOLmn<br>0 | 0: Cleared to 0 when TOMmn = 0 (master channel output mode). |

**(e) Timer output mode register m (TOMm)**

| | Bit n | |
|---|---|---|
| TOMm | TOMmn<br>0 | 0: Sets master channel output mode. |

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

**Figure 6-64. Operation Procedure When Delay Counter Function Is Used**

| | Software Operation | Hardware Status |
|---|---|---|
| TAU default setting | | Power-off status<br>(Clock supply is stopped and writing to each register is disabled.) |
| | Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1. → | Power-on status. Each channel stops operating.<br>(Clock supply is started and writing to each register is enabled.) |
| | Sets timer clock select register m (TPSm).<br>Determines clock frequencies of CKm0 to CKm3. | |
| <R> Channel default setting | Sets corresponding bit of noise filter enable register 1 (NFEN1) to 0 (OFF) or 1 (ON).<br>Sets timer mode register mn (TMRmn) (determines operation mode of channel).<br>INTTMmn output delay is set to timer data register mn (TDRmn).<br>Clears the TOEmn bit to 0 and stops operation of TOmn. | Channel stops operating.<br>(Clock is supplied and some power is consumed.) |
| <R> Operation start | Sets the TSmn bit to 1. →<br>The TSmn bit automatically returns to 0 because it is a trigger bit. | TEmn = 1, and the start trigger detection (the valid edge of the TImn pin input is detected or the TSmn bit is set to 1) wait status is set. |
| | The counter starts counting down by the next start trigger detection.<br>• Detects the TImn pin input valid edge. →<br>• Sets the TSmn bit to 1 by the software. | Value of the TDRmn register is loaded to the timer count register mn (TCRmn). |
| <R> During operation | Set value of the TDRmn register can be changed.<br>The TCRmn register can always be read.<br>The TSRmn register is not used. | The counter (TCRmn) counts down. When the count value of TCRmn reaches 0000H, the INTTMmn output is generated, and the count operation stops until the next start trigger detection (the valid edge of the TImn pin input is detected or the TSmn bit is set to 1). |
| Operation stop | The TTmn bit is set to 1. →<br>The TTmn bit automatically returns to 0 because it is a trigger bit. | TEmn = 0, and count operation stops.<br>The TCRmn register holds count value and stops. |
| TAU stop | The TAUmEN bit of the PER0 register is cleared to 0. → | Power-off status<br>All circuits are initialized and SFR of each channel is also initialized. |

Operation is resumed.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 7 (however, timer input pin (TImn), timer output pin (TOmn) : n = 0, 1, 3 to 7))

## 6.9 Simultaneous Channel Operation Function of Timer Array Unit

### 6.9.1 Operation as one-shot pulse output function

By using two channels as a set, a one-shot pulse having any delay pulse width can be generated from the signal input to the TImn pin.

The delay time and pulse width can be calculated by the following expressions.

| |
|---|
| Delay time = {Set value of TDRmn (master) + 2} × Count clock period |
| Pulse width = {Set value of TDRmp (slave)} × Count clock period |

The master channel operates in the one-count mode and counts the delays. Timer count register mn (TCRmn) of the master channel starts operating upon start trigger detection and loads the value of timer data register mn (TDRmn).

The TCRmn register counts down from the value of the TDRmn register it has loaded, in synchronization with the count clock. When TCRmn = 0000H, it outputs INTTMmn and stops counting until the next start trigger is detected.

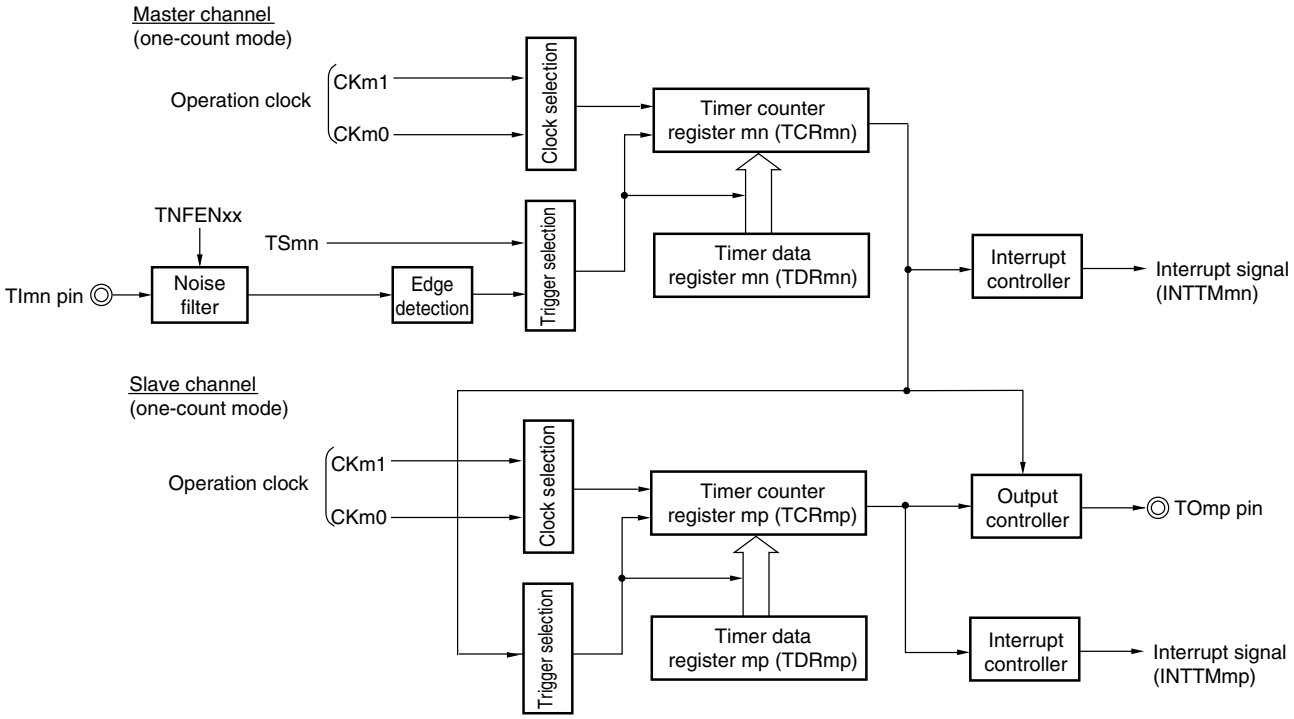
The slave channel operates in the one-count mode and counts the pulse width. The TCRmp register of the slave channel starts operation using INTTMmn of the master channel as a start trigger, and loads the value of the TDRmp register. The TCRmp register counts down from the value of The TDRmp register it has loaded, in synchronization with the count value. When count value = 0000H, it outputs INTTMmp and stops counting until the next start trigger (INTTMmn of the master channel) is detected. The output level of TOmp becomes active one count clock after generation of INTTMmn from the master channel, and inactive when TCRmp = 0000H.

Instead of using the TImn pin input, a one-shot pulse can also be output using the software operation (TSmn = 1) as a start trigger.

**Caution The timing of loading of timer data register mn (TDRmn) of the master channel is different from that of the TDRmp register of the slave channel. If the TDRmn and TDRmp registers are rewritten during operation, therefore, an illegal waveform is output. Rewrite the TDRmn register after INTTMmn is generated and the TDRmp register after INTTMmp is generated.**

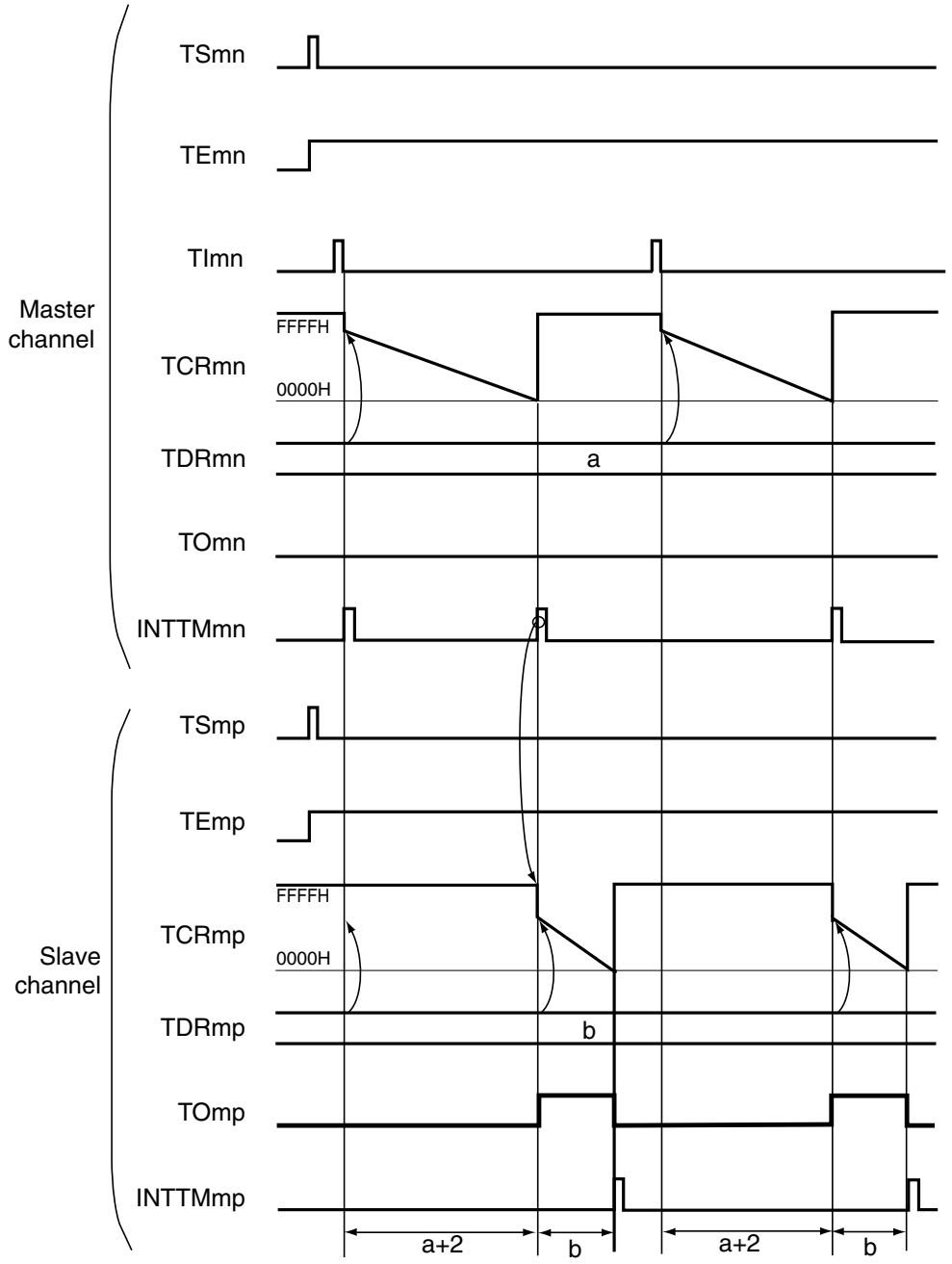
**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 4, 6)
p: Slave channel number (n < p ≤ 7)
However, timer output pin (TOmp) : p = 1, 3 to 7

**Figure 6-65. Block Diagram of Operation as One-Shot Pulse Output Function**




**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)
However, timer output pin (TOmp) : p = 1, 3 to 7

**Figure 6-66. Example of Basic Timing of Operation as One-Shot Pulse Output Function (Start Trigger: TImn Pin Input Valid Edge.)**

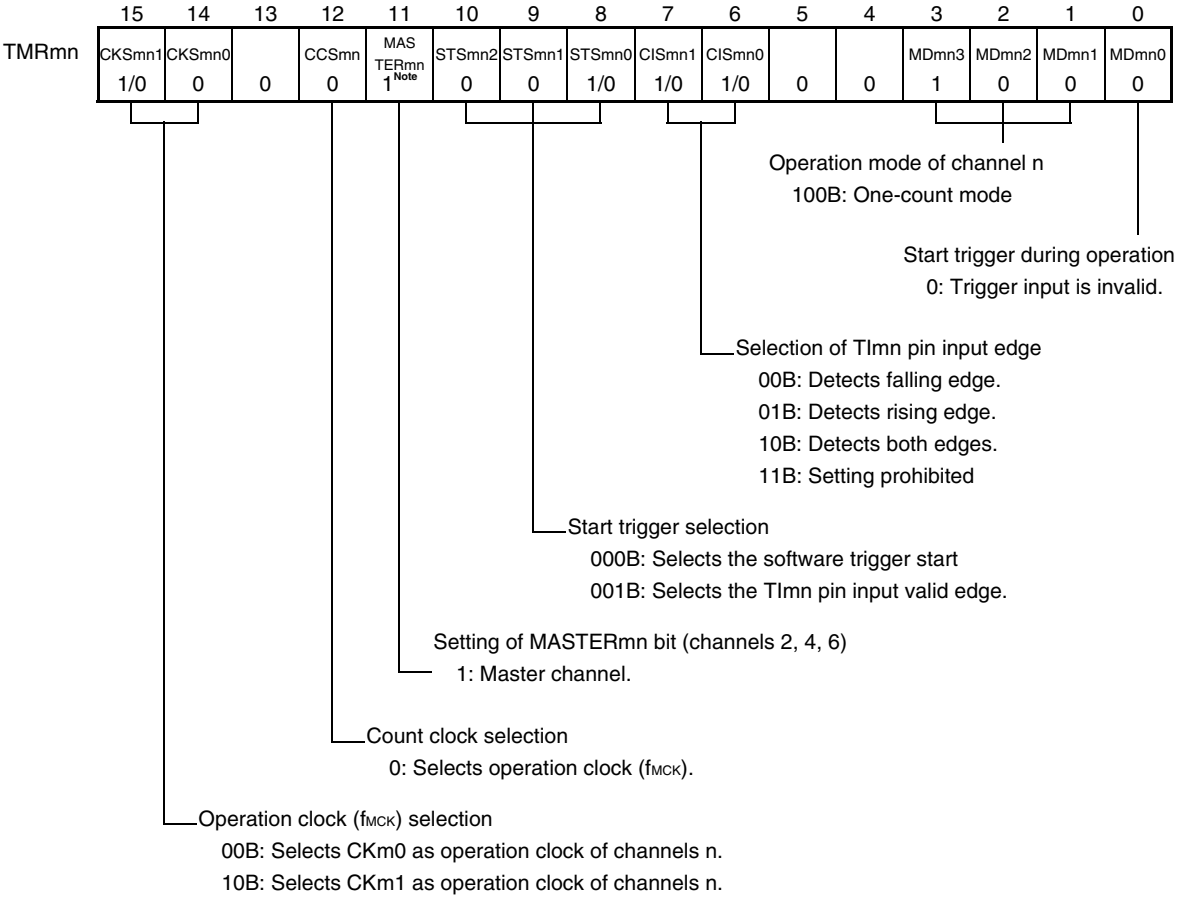



**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)
However, timer output pin (TOmp) : p = 1, 3 to 7

**2.** TSmn, TSmp: Bit n, p of timer channel start register m (TSm)
TEmn, TEmp: Bit n, p of timer channel enable status register m (TEm)
TImn, TImp: TImn and TImp pins input signal
TCRmn, TCRmp: Timer count registers mn, mp (TCRmn, TCRmp)
TDRmn, TDRmp: Timer data registers mn, mp (TDRmn, TDRmp)
TOmn, TOmp: TOmn and TOmp pins output signal

**Figure 6-67. Example of Set Contents of Registers When One-Shot Pulse Output Function Is Used (Master Channel)**

**(a) Timer mode register mn (TMRmn)**




**(b) Timer output register m (TOm)**

| | Bit n | |
|---|---|---|
| TOm | TOmn | 0: Outputs 0 from TOmn. |
| | 0 | |

**(c) Timer output enable register m (TOEm)**

| | Bit n | |
|---|---|---|
| TOEm | TOEmn | 0: Stops the TOmn output operation by counting operation. |
| | 0 | |

**(d) Timer output level register m (TOLm)**

| | Bit n | |
|---|---|---|
| TOLm | TOLmn | 0: Cleared to 0 when TOMmn = 0 (master channel output mode). |
| | 0 | |

**(e) Timer output mode register m (TOMm)**

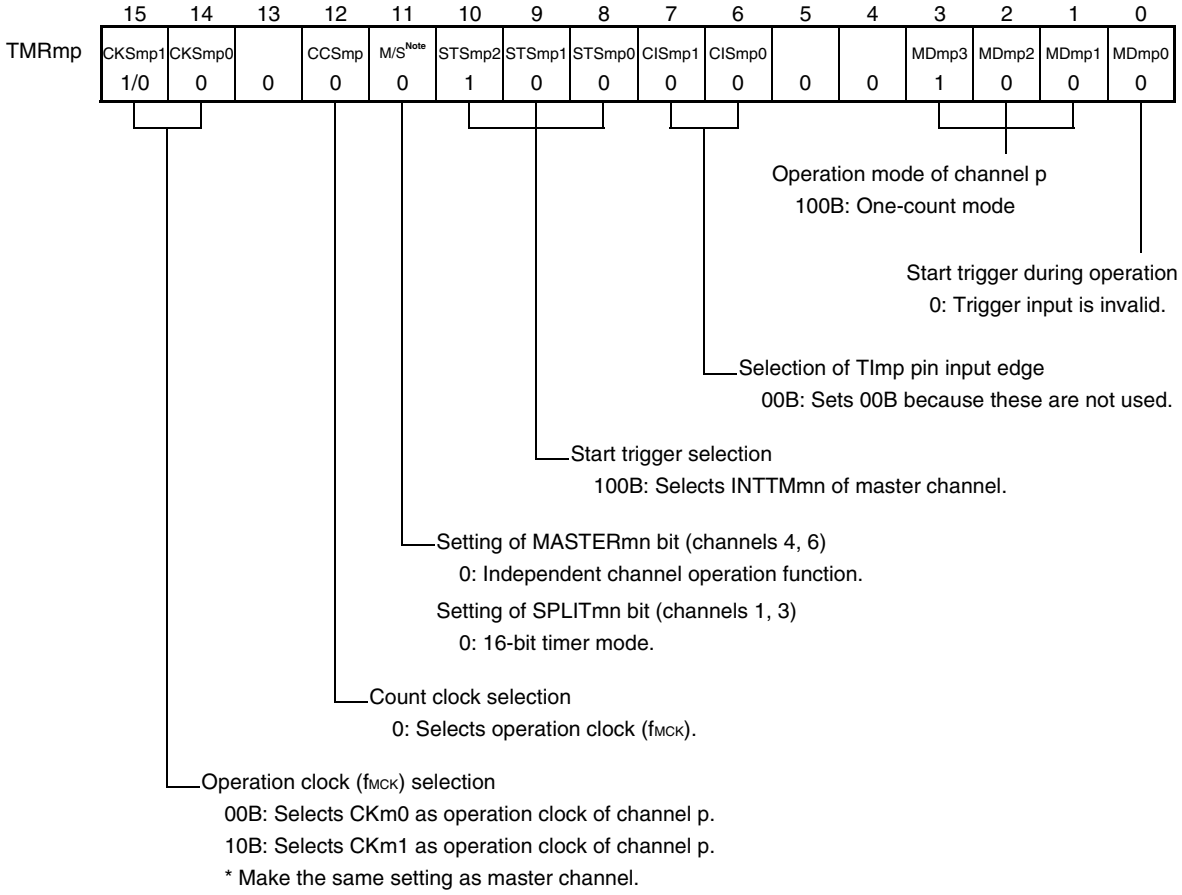
| | Bit n | |
|---|---|---|
| TOMm | TOMmn | 0: Sets master channel output mode. |
| | 0 | |

**Note** TMRm2, TMRm4, TMRm6: MASTERmn = 1
TMRm0: Fixed to 0

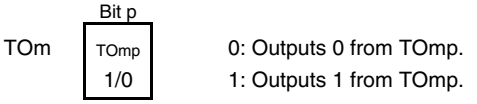
**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4, 6)

**Figure 6-68. Example of Set Contents of Registers When One-Shot Pulse Output Function Is Used (Slave Channel)**

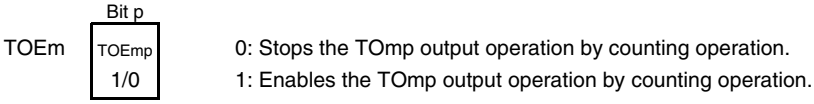
**(a) Timer mode register mp (TMRmp)**



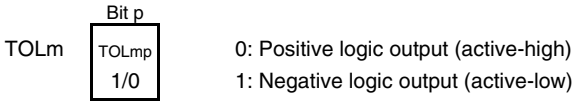

**(b) Timer output register m (TOm)**



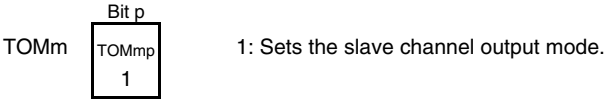

**(c) Timer output enable register m (TOEm)**




**(d) Timer output level register m (TOLm)**




**(e) Timer output mode register m (TOMm)**




**Note** TMRm4, TMRm6: MASTERmn bit
TMRm1, TMRm3: SPLITmp bit
TMRm5, TMRm7: Fixed to 0

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)
However, timer output pin (TOmp) : p = 1, 3 to 7

**Figure 6-69. Operation Procedure of One-Shot Pulse Output Function (1/2)**

| | Software Operation | Hardware Status |
|---|---|---|
| TAU default setting | | Power-off status<br>(Clock supply is stopped and writing to each register is disabled.) |
| | Sets the TAUmEN bit of peripheral enable registers 0 (PER0) to 1. → | Power-on status. Each channel stops operating.<br>(Clock supply is started and writing to each register is enabled.) |
| | Sets timer clock select register m (TPSm).<br>Determines clock frequencies of CKm0 and CKm1. | |
| <R> Channel default setting | Sets corresponding bit of noise filter enable register 1 (NFEN1) to 0 (OFF) or 1 (ON).<br>Sets timer mode register mn, mp (TMRmn, TMRmp) of two channels to be used (determines operation mode of channels).<br>An output delay is set to timer data register mn (TDRmn) of the master channel, and a pulse width is set to the TDRmp register of the slave channel. | Channel stops operating.<br>(Clock is supplied and some power is consumed.) |
| | Sets slave channel.<br>The TOMmp bit of timer output mode register m (TOMm) is set to 1 (slave channel output mode).<br>Sets the TOLmp bit.<br>Sets the TOmp bit and determines default level of the TOmp output. → | The TOmp pin goes into Hi-Z output state.<br>The TOmp default setting level is output when the port mode register is in output mode and the port register is 0. |
| | Sets the TOEmp bit to 1 and enables operation of TOmp. → | TOmp does not change because channel stops operating. |
| | Clears the port register and port mode register to 0. → | The TOmp pin outputs the TOmp set level. |

(**Note** and **Remark** are listed on the next page.)

**Figure 6-69. Operation Procedure of One-Shot Pulse Output Function (2/2)**

| | Software Operation | Hardware Status |
|---|---|---|
| <R> | Operation start | Sets the TOEmp bit (slave) to 1 (only when operation is resumed).<br>The TSmn (master) and TSmp (slave) bits of timer channel start register m (TSm) are set to 1 at the same time.<br>The TSmn and TSmp bits automatically return to 0 because they are trigger bits. | The TEmn and TEmp bits are set to 1 and the master channel enters the start trigger detection (the valid edge of the TImn pin input is detected or the TSmn bit of the master channel is set to 1) wait status.<br>Counter stops operating. |
| | | Count operation of the master channel is started by start trigger detection of the master channel.<br>• Detects the TImn pin input valid edge.<br>• Sets the TSmn bit of the master channel to 1 by software[Note].<br>**Note** Do not set the TSmn bit of the slave channel to 1. | Master channel starts counting. |
| <R> | During operation | Set values of only the CISmn1 and CISmn0 bits of the TMRmn register can be changed.<br>Set values of the TMRmp, TDRmn, TDRmp registers, TOMmn, TOMmp, TOLmn, and TOLmp bits cannot be changed.<br>The TCRmn and TCRmp registers can always be read.<br>The TSRmn and TSRmp registers are not used.<br>Set values of the TOm and TOEm registers by slave channel can be changed. | Master channel loads the value of the TDRmn register to timer count register mn (TCRmn) by the start trigger detection (the valid edge of the TImn pin input is detected or the TSmn bit of the master channel is set to 1), and the counter starts counting down. When the count value reaches TCRmn = 0000H, the INTTMmn output is generated, and the counter stops until the next start trigger detection.<br>The slave channel, triggered by INTTMmn of the master channel, loads the value of the TDRmp register to the TCRmp register, and the counter starts counting down.<br>The output level of TOmp becomes active one count clock after generation of INTTMmn from the master channel. It becomes inactive when TCRmp = 0000H, and the counting operation is stopped.<br>After that, the above operation is repeated. |
| | Operation stop | The TTmn (master) and TTmp (slave) bits are set to 1 at the same time.<br>The TTmn and TTmp bits automatically return to 0 because they are trigger bits. | TEmn, TEmp = 0, and count operation stops.<br>The TCRmn and TCRmp registers hold count value and stop.<br>The TOmp output is not initialized but holds current status. |
| | | The TOEmp bit of slave channel is cleared to 0 and value is set to the TOmp bit. | The TOmp pin outputs the TOmp set level. |
| | TAU stop | To hold the TOmp pin output level<br>Clears the TOmp bit to 0 after the value to be held is set to the port register.<br>When holding the TOmp pin output level is not necessary<br>Setting not required. | The TOmp pin output level is held by port function. |
| | | The TAUmEN bit of the PER0 register is cleared to 0. | Power-off status<br>All circuits are initialized and SFR of each channel is also initialized.<br>(The TOmp bit is cleared to 0 and the TOmp pin is set to port mode.) |

Operation is resumed.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)
However, timer output pin (TOmp) : p = 1, 3 to 7

### 6.9.2 Operation as PWM function

Two channels can be used as a set to generate a pulse of any period and duty factor.

The period and duty factor of the output pulse can be calculated by the following expressions.

| |
|---|
| Pulse period = {Set value of TDRmn (master) + 1} × Count clock period |
| Duty factor [%] = {Set value of TDRmp (slave)}/{Set value of TDRmn (master) + 1} × 100 |
| 0% output: Set value of TDRmp (slave) = 0000H |
| 100% output: Set value of TDRmp (slave) ≥ {Set value of TDRmn (master) + 1} |

**Remark** The duty factor exceeds 100% if the set value of TDRmp (slave) > (set value of TDRmn (master) + 1), it summarizes to 100% output.

The master channel operates in the interval timer mode. If the channel start trigger bit (TSmn) of timer channel start register m (TSm) is set to 1, an interrupt (INTTMmn) is output, the value set to timer data register mn (TDRmn) is loaded to timer count register mn (TCRmn), and the counter counts down in synchronization with the count clock. When the counter reaches 0000H, INTTMmn is output, the value of the TDRmn register is loaded again to the TCRmn register, and the counter counts down. This operation is repeated until the channel stop trigger bit (TTmn) of timer channel stop register m (TTm) is set to 1.

If two channels are used to output a PWM waveform, the period until the master channel counts down to 0000H is the PWM output (TOmp) cycle.

The slave channel operates in one-count mode. By using INTTMmn from the master channel as a start trigger, the TCRmp register loads the value of the TDRmp register and the counter counts down to 0000H. When the counter reaches 0000H, it outputs INTTMmp and waits until the next start trigger (INTTMmn from the master channel) is generated.

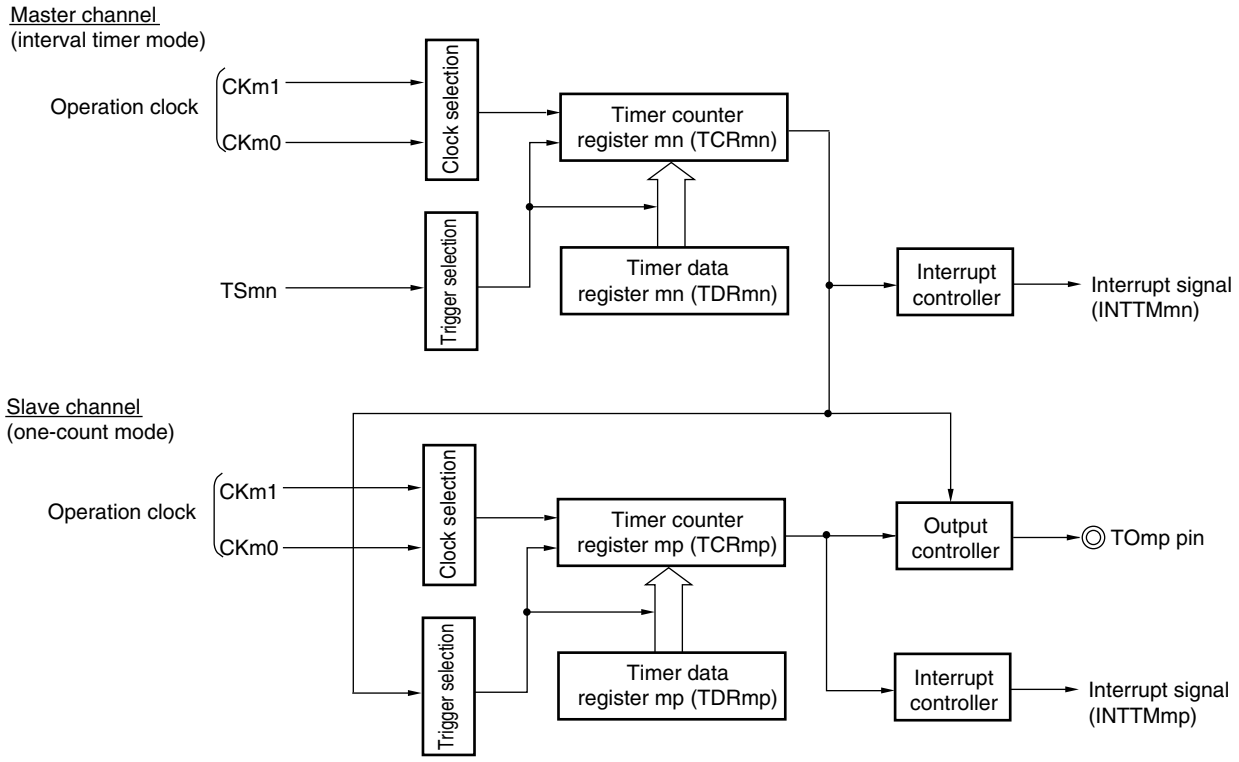
If two channels are used to output a PWM waveform, the period until the slave channel counts down to 0000H is the PWM output (TOmp) duty.

PWM output (TOmp) goes to the active level one clock after the master channel generates INTTMmn and goes to the inactive level when the TCRmp register of the slave channel becomes 0000H.

**Caution To rewrite both timer data register mn (TDRmn) of the master channel and the TDRmp register of the slave channel, a write access is necessary two times. The timing at which the values of the TDRmn and TDRmp registers are loaded to the TCRmn and TCRmp registers is upon occurrence of INTTMmn of the master channel. Thus, when rewriting is performed split before and after occurrence of INTTMmn of the master channel, the TOmp pin cannot output the expected waveform. To rewrite both the TDRmn register of the master and the TDRmp register of the slave, therefore, be sure to rewrite both the registers immediately after INTTMmn is generated from the master channel.**

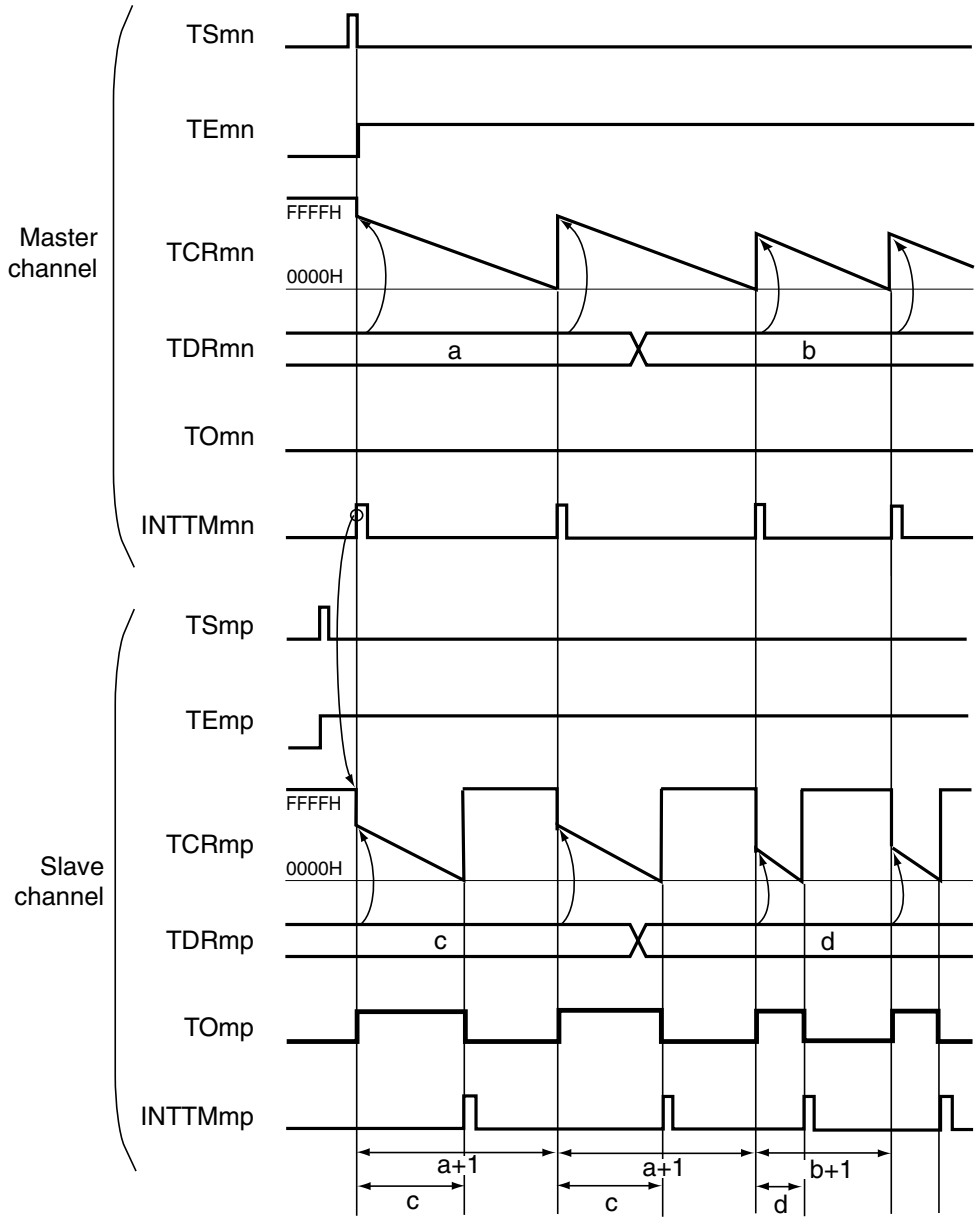
**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7 )
However, timer output pin (TOmp) : p = 1, 3 to 7

**Figure 6-70. Block Diagram of Operation as PWM Function**




**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)
However, timer output pin (TOmp) : p = 1, 3 to 7

**Figure 6-71. Example of Basic Timing of Operation as PWM Function**

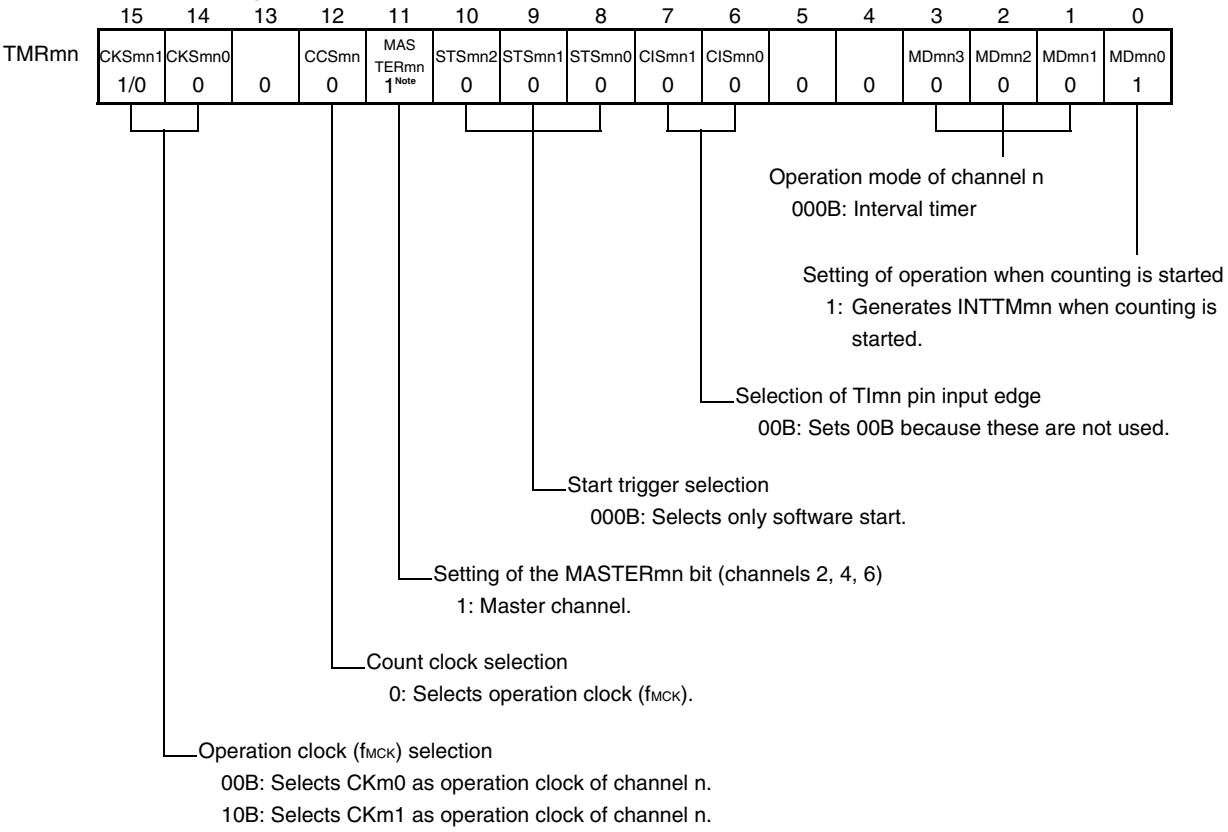



**Remark 1.** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)
However, timer output pin (TOmp) : p = 1, 3 to 7

**2.** TSmn, TSmp: Bit n, p of timer channel start register m (TSm)
TEmn, TEmp: Bit n, p of timer channel enable status register m (TEm)
TCRmn, TCRmp: Timer count registers mn, mp (TCRmn, TCRmp)
TDRmn, TDRmp: Timer data registers mn, mp (TDRmn, TDRmp)
TOmn, TOmp: TOmn and TOmp pins output signal

**Figure 6-72. Example of Set Contents of Registers When PWM Function (Master Channel) Is Used**

**(a) Timer mode register mn (TMRmn)**




<R>

**(b) Timer output register m (TOm)**

| | Bit n | |
|---|---|---|
| TOm | TOmn | 0: Outputs 0 from TOmn. |
| | 0 | |

**(c) Timer output enable register m (TOEm)**

| | Bit n | |
|---|---|---|
| TOEm | TOEmn | 0: Stops the TOmn output operation by counting operation. |
| | 0 | |

**(d) Timer output level register m (TOLm)**

| | Bit n | |
|---|---|---|
| TOLm | TOLmn | 0: Cleared to 0 when TOMmn = 0 (master channel output mode). |
| | 0 | |

**(e) Timer output mode register m (TOMm)**

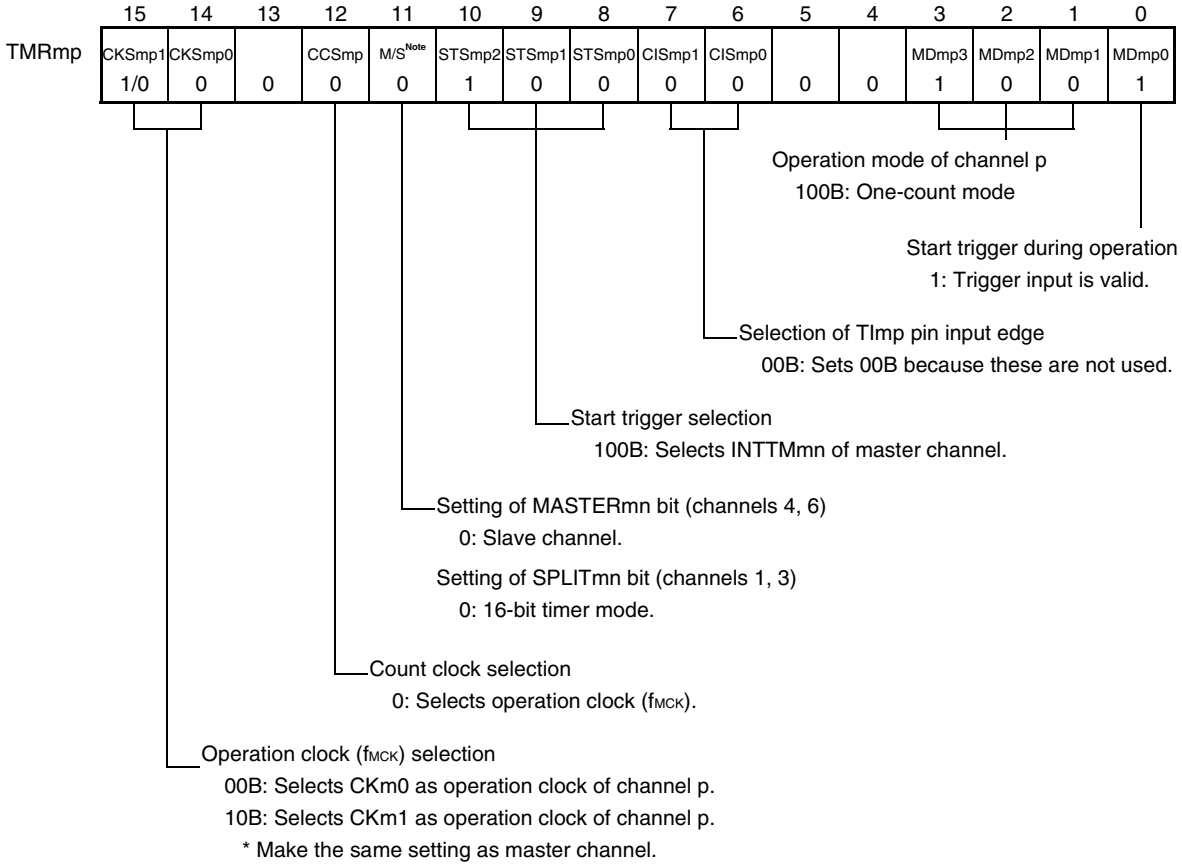
| | Bit n | |
|---|---|---|
| TOMm | TOMmn | 0: Sets master channel output mode. |
| | 0 | |

<R> **Note** TMRm2, TMRm4, TMRm6: MASTERmn = 1
TMRm0 :Fixed to 0

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4, 6)

**Figure 6-73. Example of Set Contents of Registers When PWM Function (Slave Channel) Is Used**

**(a) Timer mode register mp (TMRmp)**




**(b) Timer output register m (TOm)**

| | Bit p | |
|---|---|---|
| TOm | TOmp<br>1/0 | 0: Outputs 0 from TOmp.<br>1: Outputs 1 from TOmp. |

**(c) Timer output enable register m (TOEm)**

| | Bit p | |
|---|---|---|
| TOEm | TOEmp<br>1/0 | 0: Stops the TOmp output operation by counting operation.<br>1: Enables the TOmp output operation by counting operation. |

**(d) Timer output level register m (TOLm)**

| | Bit p | |
|---|---|---|
| TOLm | TOLmp<br>1/0 | 0: Positive logic output (active-high)<br>1: Negative logic output (active-low) |

**(e) Timer output mode register m (TOMm)**

| | Bit p | |
|---|---|---|
| TOMm | TOMmp<br>1 | 1: Sets the slave channel output mode. |

**Note** TMRm4, TMRm6: MASTERmn bit
TMRm1, TMRm3: SPLITmp bit
TMRm5, TMRm7: Fixed to 0

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)
However, timer output pin (TOmp) : p = 1, 3 to 7

**Figure 6-74. Operation Procedure When PWM Function Is Used (1/2)**

| | Software Operation | Hardware Status |
|---|---|---|
| TAU default setting | | Power-off status<br>(Clock supply is stopped and writing to each register is disabled.) |
| | Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1. → | Power-on status. Each channel stops operating.<br>(Clock supply is started and writing to each register is enabled.) |
| | Sets timer clock select register m (TPSm).<br>Determines clock frequencies of CKm0 and CKm1. | |
| Channel default setting | Sets timer mode registers mn, mp (TMRmn, TMRmp) of two channels to be used (determines operation mode of channels).<br>An interval (period) value is set to timer data register mn (TDRmn) of the master channel, and a duty factor is set to the TDRmp register of the slave channel. | Channel stops operating.<br>(Clock is supplied and some power is consumed.) |
| | Sets slave channel.<br>The TOMmp bit of timer output mode register m (TOMm) is set to 1 (slave channel output mode).<br>Sets the TOLmp bit.<br>Sets the TOmp bit and determines default level of the TOmp output. → | The TOmp pin goes into Hi-Z output state.<br>The TOmp default setting level is output when the port mode register is in output mode and the port register is 0. |
| | Sets the TOEmp bit to 1 and enables operation of TOmp. → | TOmp does not change because channel stops operating. |
| | Clears the port register and port mode register to 0. → | The TOmp pin outputs the TOmp set level. |

**Figure 6-74. Operation Procedure When PWM Function Is Used (2/2)**

| | Software Operation | Hardware Status |
|---|---|---|
| Operation start | Sets the TOEmp bit (slave) to 1 (only when operation is resumed).<br>The TSmn (master) and TSmp (slave) bits of timer channel start register m (TSm) are set to 1 at the same time. → | TEmn = 1, TEmp = 1<br>→ When the master channel starts counting, INTTMmn is generated. Triggered by this interrupt, the slave channel also starts counting. |
| | The TSmn and TSmp bits automatically return to 0 because they are trigger bits. | |
| During operation | Set values of the TMRmn and TMRmp registers, TOMmn, TOMmp, TOLmn, and TOLmp bits cannot be changed.<br>Set values of the TDRmn and TDRmp registers can be changed after INTTMmn of the master channel is generated.<br>The TCRmn and TCRmp registers can always be read.<br>The TSRmn and TSRmp registers are not used. | The counter of the master channel loads the TDRmn register value to timer count register mn (TCRmn), and counts down. When the count value reaches TCRmn = 0000H, INTTMmn output is generated. At the same time, the value of the TDRmn register is loaded to the TCRmn register, and the counter starts counting down again.<br>At the slave channel, the value of the TDRmp register is loaded to the TCRmp register, triggered by INTTMmn of the master channel, and the counter starts counting down.<br>The output level of TOmp becomes active one count clock after generation of the INTTMmn output from the master channel. It becomes inactive when TCRmp = 0000H, and the counter operation is stopped.<br>After that, the above operation is repeated. |
| Operation stop | The TTmn (master) and TTmp (slave) bits are set to 1 at the same time. → | TEmn, TEmp = 0, and count operation stops. |
| | The TTmn and TTmp bits automatically return to 0 because they are trigger bits. | The TCRmn and TCRmp registers hold count value and stop.<br>The TOmp output is not initialized but holds current status. |
| | The TOEmp bit of slave channel is cleared to 0 and value is set to the TOmp bit. → | The TOmp pin outputs the TOmp set level. |
| TAU stop | To hold the TOmp pin output level<br>Clears the TOmp bit to 0 after the value to be held is set to the port register. → | The TOmp pin output level is held by port function. |
| | When holding the TOmp pin output level is not necessary<br>Setting not required. | |
| | The TAUmEN bit of the PER0 register is cleared to 0. → | Power-off status<br>All circuits are initialized and SFR of each channel is also initialized.<br>(The TOmp bit is cleared to 0 and the TOmp pin is set to port mode.) |

Operation is resumed.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4, 6)
p: Slave channel number (n < p ≤ 7)
However, timer output pin (TOmp) : p = 1, 3 to 7

### 6.9.3 Operation as multiple PWM output function

By extending the PWM function and using multiple slave channels, many PWM waveforms with different duty values can be output.

For example, when using two slave channels, the period and duty factor of an output pulse can be calculated by the following expressions.

| |
|---|
| Pulse period = {Set value of TDRmn (master) + 1} × Count clock period<br>Duty factor 1 [%] = {Set value of TDRmp (slave 1)}/{Set value of TDRmn (master) + 1} × 100<br>Duty factor 2 [%] = {Set value of TDRmq (slave 2)}/{Set value of TDRmn (master) + 1} × 100 |

**Remark** Although the duty factor exceeds 100% if the set value of TDRmp (slave 1) > {set value of TDRmn (master) + 1} or if the {set value of TDRmq (slave 2)} > {set value of TDRmn (master) + 1}, it is summarized into 100% output.

Timer count register mn (TCRmn) of the master channel operates in the interval timer mode and counts the periods.

The TCRmp register of the slave channel 1 operates in one-count mode, counts the duty factor, and outputs a PWM waveform from the TOmp pin. The TCRmp register loads the value of timer data register mp (TDRmp), using INTTMmn of the master channel as a start trigger, and starts counting down. When TCRmp = 0000H, TCRmp outputs INTTMmp and stops counting until the next start trigger (INTTMmn of the master channel) has been input. The output level of TOmp becomes active one count clock after generation of INTTMmn from the master channel, and inactive when TCRmp = 0000H.

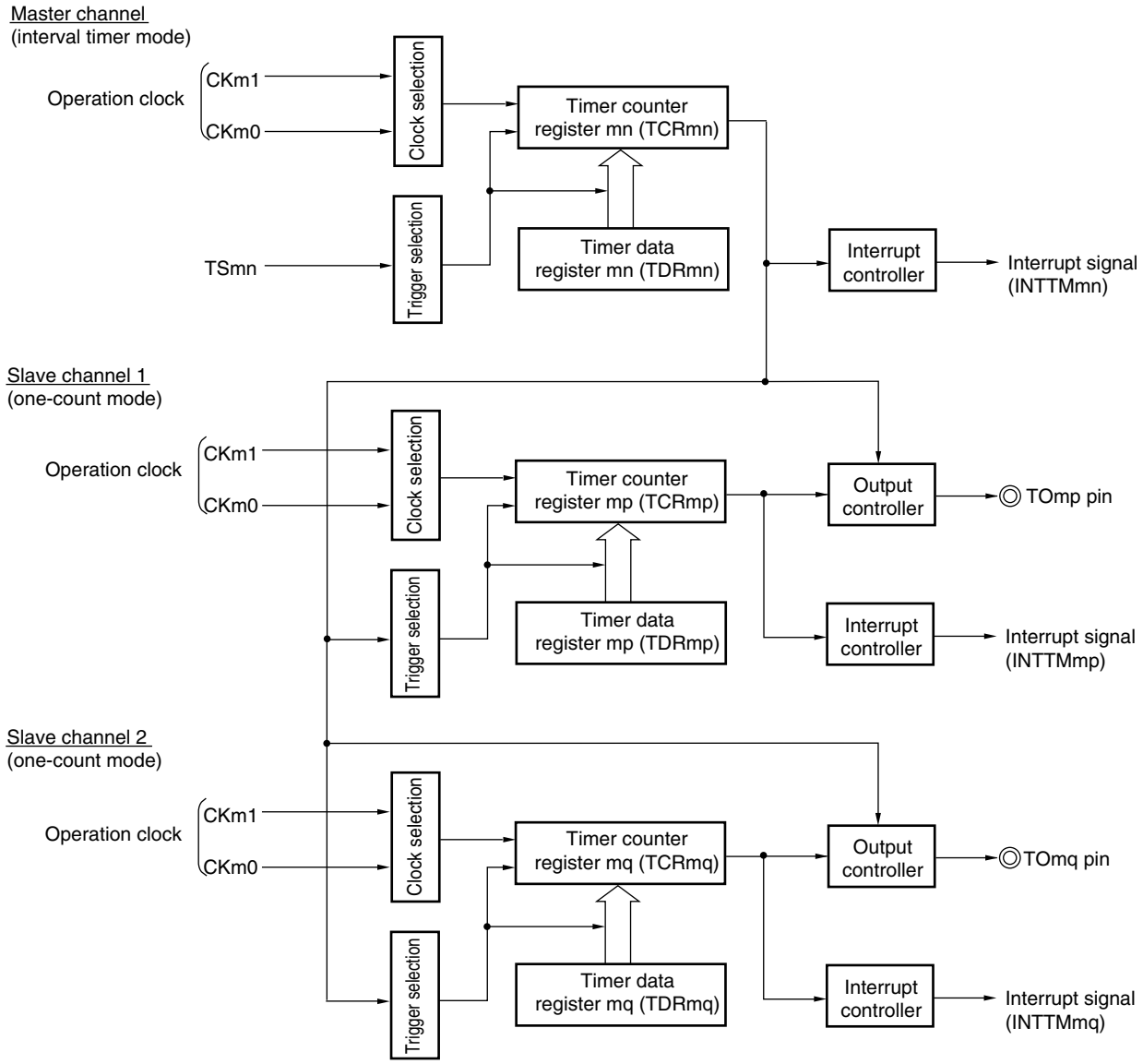
In the same way as the TCRmp register of the slave channel 1, the TCRmq register of the slave channel 2 operates in one-count mode, counts the duty factor, and outputs a PWM waveform from the TOmq pin. The TCRmq register loads the value of the TDRmq register, using INTTMmn of the master channel as a start trigger, and starts counting down. When TCRmq = 0000H, the TCRmq register outputs INTTMmq and stops counting until the next start trigger (INTTMmn of the master channel) has been input. The output level of TOmq becomes active one count clock after generation of INTTMmn from the master channel, and inactive when TCRmq = 0000H.

When channel 0 is used as the master channel as above, up to six types of PWM signals can be output at the same time.

**Caution To rewrite both timer data register mn (TDRmn) of the master channel and the TDRmp register of the slave channel 1, write access is necessary at least twice. Since the values of the TDRmn and TDRmp registers are loaded to the TCRmn and TCRmp registers after INTTMmn is generated from the master channel, if rewriting is performed separately before and after generation of INTTMmn from the master channel, the TOmp pin cannot output the expected waveform. To rewrite both the TDRmn register of the master and the TDRmp register of the slave, be sure to rewrite both the registers immediately after INTTMmn is generated from the master channel (This applies also to the TDRmq register of the slave channel 2).**

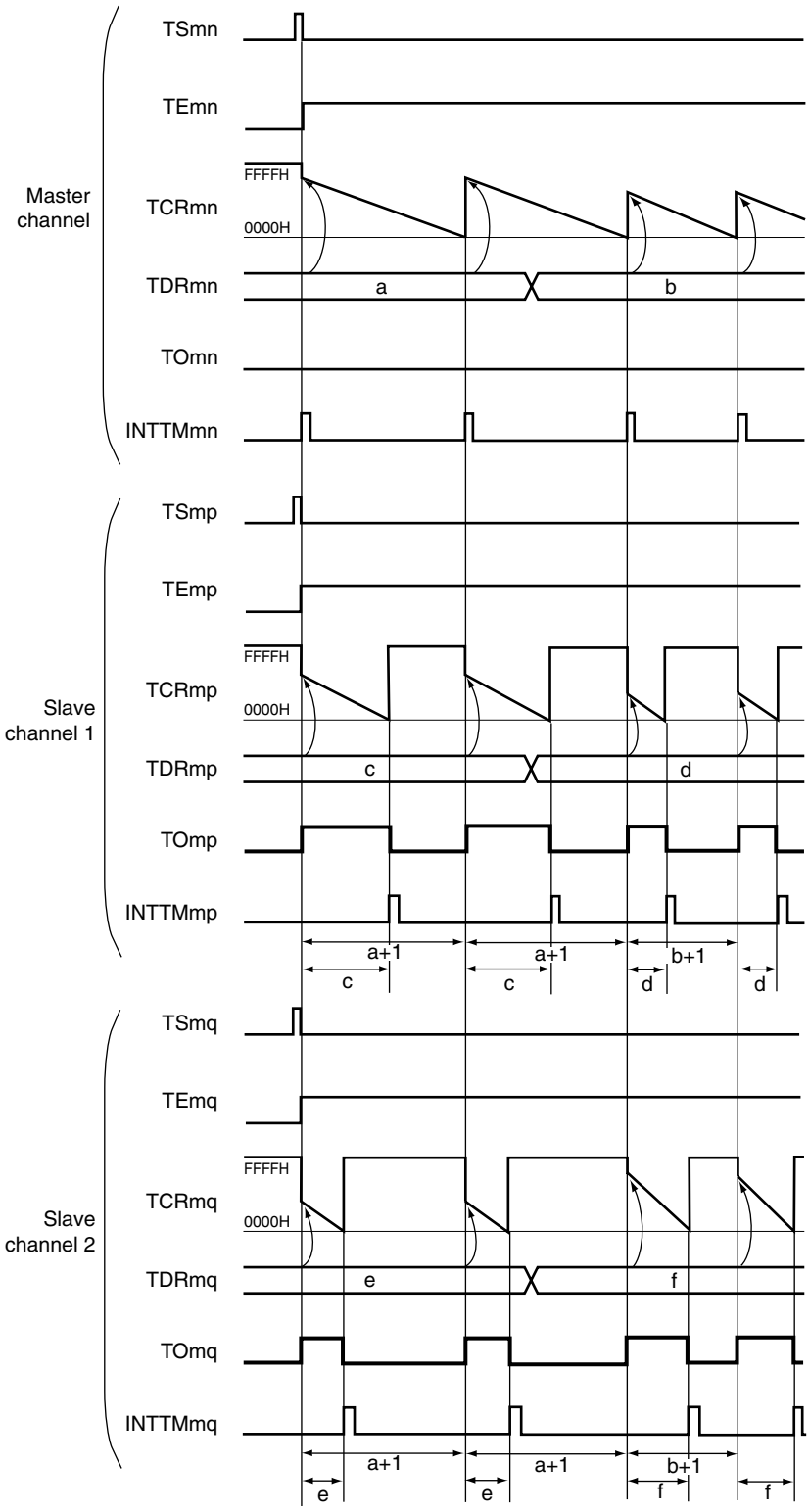
**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4)
p: Slave channel number 1, q: Slave channel number 2
$n < p < q \leq 7$
However, timer output pin (TOmp, TOmq) : p = 1, 3 to 6, q = 3 to 7

**Figure 6-75. Block Diagram of Operation as Multiple PWM Output Function (output two types of PWMs)**




**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4)
p: Slave channel number 1, q: Slave channel number 2
$n < p < q \leq 7$
However, timer output pin (TOmp, TOmq) : p = 1, 3 to 6, q = 3 to 7

**Figure 6-76. Example of Basic Timing of Operation as Multiple PWM Output Function (Output two types of PWMs)**



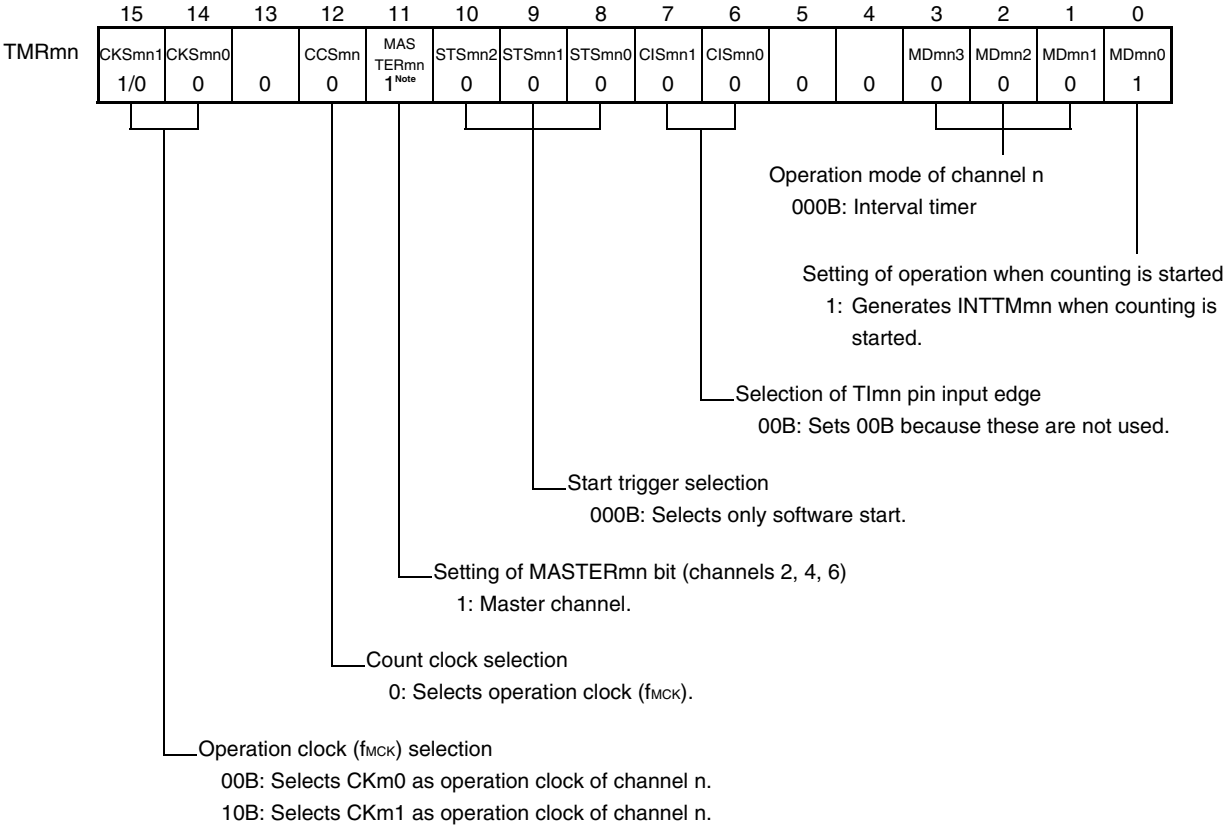

(**Remark** is listed on the next page.)

**Remarks1.** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4)
p: Slave channel number 1, q: Slave channel number 2
$n < p < q \leq 7$
However, timer output pin (TOmp, TOmq) : p = 1, 3 to 6, q = 3 to 7

**2.** TSmn, TSmp, TSmq: Bit n, p, q of timer channel start register m (TSm)
TEmn, TEmp, TEmq: Bit n, p, q of timer channel enable status register m (TEm)
TCRmn, TCRmp, TCRmq: Timer count registers mn, mp, mq (TCRmn, TCRmp, TCRmq)
TDRmn, TDRmp, TDRmq: Timer data registers mn, mp, mq (TDRmn, TDRmp, TDRmq)
TOmn, TOmp, TOmq: TOmn, TOmp, and TOmq pins output signal

**Figure 6-77. Example of Set Contents of Registers When Multiple PWM Output Function (Master Channel) Is Used**

**(a) Timer mode register mn (TMRmn)**




<R>

**(b) Timer output register m (TOm)**

| | Bit n | |
|---|---|---|
| TOm | TOmn<br>0 | 0: Outputs 0 from TOmn. |

**(c) Timer output enable register m (TOEm)**

| | Bit n | |
|---|---|---|
| TOEm | TOEmn<br>0 | 0: Stops the TOmn output operation by counting operation. |

**(d) Timer output level register m (TOLm)**

| | Bit n | |
|---|---|---|
| TOLm | TOLmn<br>0 | 0: Cleared to 0 when TOMmn = 0 (master channel output mode). |

**(e) Timer output mode register m (TOMm)**

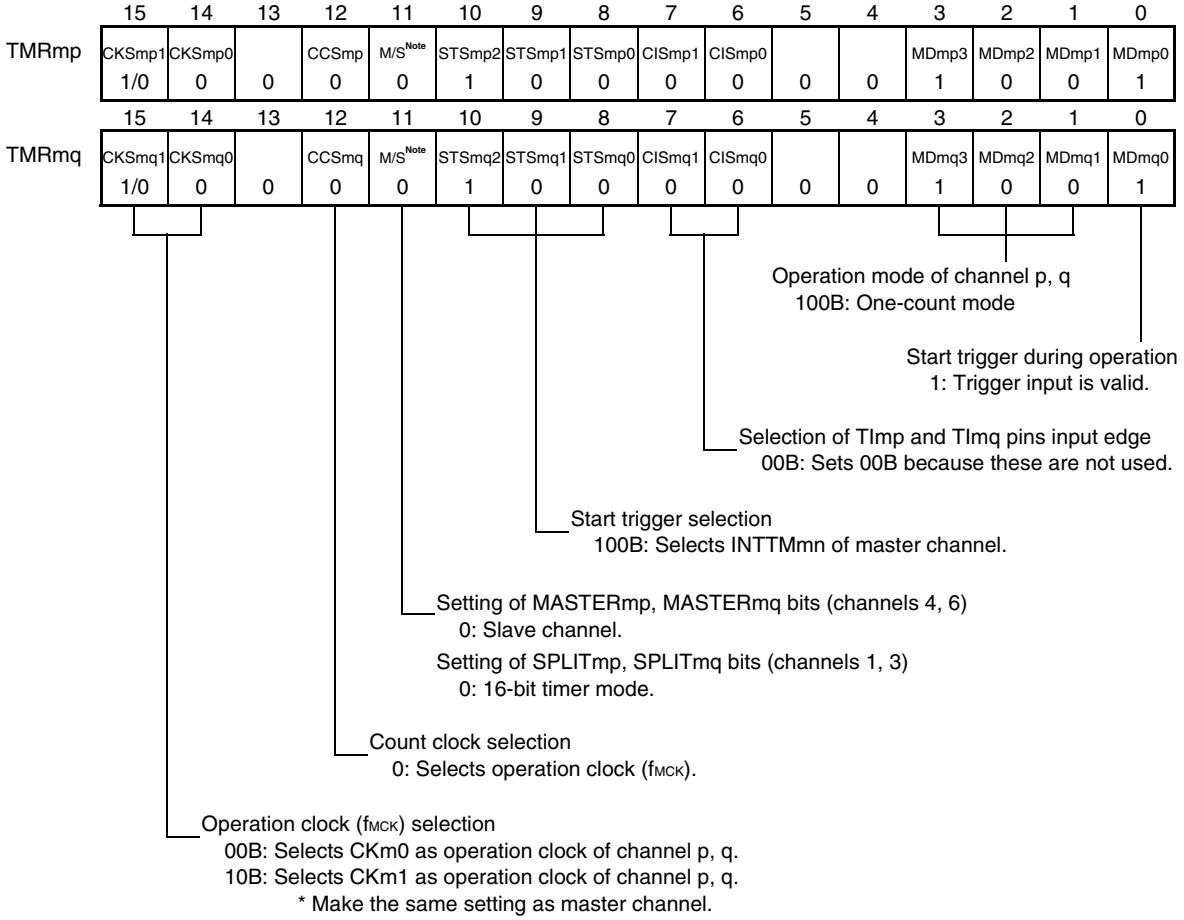
| | Bit n | |
|---|---|---|
| TOMm | TOMmn<br>0 | 0: Sets master channel output mode. |

<R> **Note** TMRm2, TMRm4, TMRm6: MASTERmn = 1
TMRm0 :Fixed to 0

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4)

**Figure 6-78. Example of Set Contents of Registers**
**When Multiple PWM Output Function (Slave Channel) Is Used (output two types of PWMs)**

**(a) Timer mode register mp, mq (TMRmp, TMRmq)**




**(b) Timer output register m (TOm)**

| | Bit q | Bit p | |
|---|---|---|---|
| TOm | TOmq | TOmp | 0: Outputs 0 from TOmp or TOmq.<br>1: Outputs 1 from TOmp or TOmq. |
| | 1/0 | 1/0 | |

**(c) Timer output enable register m (TOEm)**

| | Bit q | Bit p | |
|---|---|---|---|
| TOEm | TOEmq | TOEmp | 0: Stops the TOmp or TOmq output operation by counting operation.<br>1: Enables the TOmp or TOmq output operation by counting operation. |
| | 1/0 | 1/0 | |

**(d) Timer output level register m (TOLm)**

| | Bit q | Bit p | |
|---|---|---|---|
| TOLm | TOLmq | TOLmp | 0: Positive logic output (active-high)<br>1: Negative logic output (active-low) |
| | 1/0 | 1/0 | |

**(e) Timer output mode register m (TOMm)**

| | Bit q | Bit p | |
|---|---|---|---|
| TOMm | TOMmq | TOMmp | 1: Sets the slave channel output mode. |
| | 1 | 1 | |

**Note** TMRm4, TMRm6: MASTERmp, MASTERmq bit
TMRm1, TMRm3: SPLITmp, SPLITmq bit
TMRm5, TMRm7: Fixed to 0

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4)
p: Slave channel number 1, q: Slave channel number 2
$n < p < q \leq 7$
However, timer output pin (TOmp, TOmq) : p = 1, 3 to 6, q = 3 to 7

**Figure 6-79. Operation Procedure When Multiple PWM Output Function Is Used (1/2)**

| | Software Operation | Hardware Status |
|---|---|---|
| TAU default setting | | Power-off status<br>(Clock supply is stopped and writing to each register is disabled.) |
| | Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1. → | Power-on status. Each channel stops operating.<br>(Clock supply is started and writing to each register is enabled.) |
| | Sets timer clock select register m (TPSm).<br>Determines clock frequencies of CKm0 and CKm1. | |
| Channel default setting | Sets timer mode registers mn, mp, 0q (TMRmn, TMRmp, TMRmq) of each channel to be used (determines operation mode of channels).<br>An interval (period) value is set to timer data register mn (TDRmn) of the master channel, and a duty factor is set to the TDRmp and TDRmq registers of the slave channels. | Channel stops operating.<br>(Clock is supplied and some power is consumed.) |
| | Sets slave channels.<br>The TOMmp and TOMmq bits of timer output mode register m (TOMm) are set to 1 (slave channel output mode).<br>Clears the TOLmp and TOLmq bits to 0.<br>Sets the TOmp and TOmq bits and determines default level of the TOmp and TOmq outputs. → | The TOmp and TOmq pins go into Hi-Z output state.<br>The TOmp and TOmq default setting levels are output when the port mode register is in output mode and the port register is 0. |
| | Sets the TOEmp and TOEmq bits to 1 and enables operation of TOmp and TOmq. → | TOmp and TOmq do not change because channels stop operating. |
| | Clears the port register and port mode register to 0. → | The TOmp and TOmq pins output the TOmp and TOmq set levels. |

(**Note** and **Remark** are listed on the next page.)

**Figure 6-79. Operation Procedure When Multiple PWM Output Function Is Used (2/2)**

| | Software Operation | Hardware Status |
|---|---|---|
| Operation start | (Sets the TOEmp and TOEmq (slave) bits to 1 only when resuming operation.)<br>The TSmn bit (master), and TSmp and TSmq (slave) bits of timer channel start register m (TSm) are set to 1 at the same time.<br>The TSmn, TSmp, and TSmq bits automatically return to 0 because they are trigger bits. | → TEmn = 1, TEmp, TEmq = 1<br>When the master channel starts counting, INTTMmn is generated. Triggered by this interrupt, the slave channel also starts counting. |
| During operation | Set values of the TMRmn, TMRmp, TMRmq registers, TOMmn, TOMmp, TOMmq, TOLmn, TOLmp, and TOLmq bits cannot be changed.<br>Set values of the TDRmn, TDRmp, and TDRmq registers can be changed after INTTMmn of the master channel is generated.<br>The TCRmn, TCRmp, and TCRmq registers can always be read.<br>The TSRmn, TSRmp, and TSR0q registers are not used. | The counter of the master channel loads the TDRmn register value to timer count register mn (TCRmn) and counts down. When the count value reaches TCRmn = 0000H, INTTMmn output is generated. At the same time, the value of the TDRmn register is loaded to the TCRmn register, and the counter starts counting down again.<br>At the slave channel 1, the values of the TDRmp register are transferred to the TCRmp register, triggered by INTTMmn of the master channel, and the counter starts counting down. The output levels of TOmp become active one count clock after generation of the INTTMmn output from the master channel. It becomes inactive when TCRmp = 0000H, and the counting operation is stopped.<br>At the slave channel 2, the values of the TDRmq register are transferred to TCRmq regster, triggered by INTTMmn of the master channel, and the counter starts counting down. The output levels of TOmq become active one count clock after generation of the INTTMmn output from the master channel. It becomes inactive when TCRmq = 0000H, and the counting operation is stopped.<br>After that, the above operation is repeated. |
| Operation stop | The TTmn bit (master), TTmp, and TTmq (slave) bits are set to 1 at the same time.<br>The TTmn, TTmp, and TTmq bits automatically return to 0 because they are trigger bits. | → TEmn, TEmp, TEmq = 0, and count operation stops.<br>The TCRmn, TCRmp, and TCRmq registers hold count value and stop.<br>The TOmp and TOmq output are not initialized but hold current status. |
| | The TOEmp and TOEmq bits of slave channels are cleared to 0 and value is set to the TOmp and TOmq bits. | → The TOmp and TOmq pins output the TOmp and TOmq set levels. |
| TAU stop | To hold the TOmp and TOmq pin output levels<br>Clears the TOmp and TOmq bits to 0 after the value to be held is set to the port register.<br>When holding the TOmp and TOmq pin output levels are not necessary<br>Setting not required | → The TOmp and TOmq pin output levels are held by port function. |
| | The TAUmEN bit of the PER0 register is cleared to 0. | → Power-off status<br>All circuits are initialized and SFR of each channel is also initialized.<br>(The TOmp and TOmq bits are cleared to 0 and the TOmp and TOmq pins are set to port mode.) |

Operation is resumed.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2, 4)
p: Slave channel number 1, q: Slave channel number 2
$n < p < q \leq 7$
However, timer output pin (TOmp, TOmq) : p = 1, 3 to 6, q = 3 to 7

## 6.10 Cautions When Using Timer Array Unit

### 6.10.1 Cautions when using timer output

Depends on products, a pin is assigned a timer output and other alternate functions. In this case, outputs of the other alternate functions must be set in initial status.

<R> For details, see **4.5 Register Settings When Using Alternate Function**.

# CHAPTER 7 REAL-TIME CLOCK

## 7.1 Functions of Real-time Clock

The real-time clock has the following features.

- Having counters of year, month, week, day, hour, minute, and second, and can count up to 99 years.
- Constant-period interrupt function (period: 0.5 seconds, 1 second, 1 minute, 1 hour, 1 day, 1 month)
- Alarm interrupt function (alarm: week, hour, minute)
- Pin output function of 1 Hz (48 and 64-pin products only)

<R> The real-time clock interrupt signal (INTRTC) can be utilized for wakeup from STOP mode and triggering an A/D converter's SNOOZE mode.

**Caution The count of year, month, week, day, hour, minutes and second can only be performed when a subsystem clock ($f_{SUB}$ = 32.768 kHz) is selected as the operation clock of the real-time clock. When the low-speed oscillation clock ($f_{IL}$ = 15 kHz (TYP.)) is selected, only the constant-period interrupt function is available. The 25- to 32-pin products have the constant-period interrupt function only, because these products have no subsystem clock.**
**However, the constant-period interrupt interval when $f_{IL}$ is selected will be calculated with the constant-period (the value selected with RTCC0 register) × $f_{SUB}/f_{IL}$.**

## 7.2 Configuration of Real-time Clock

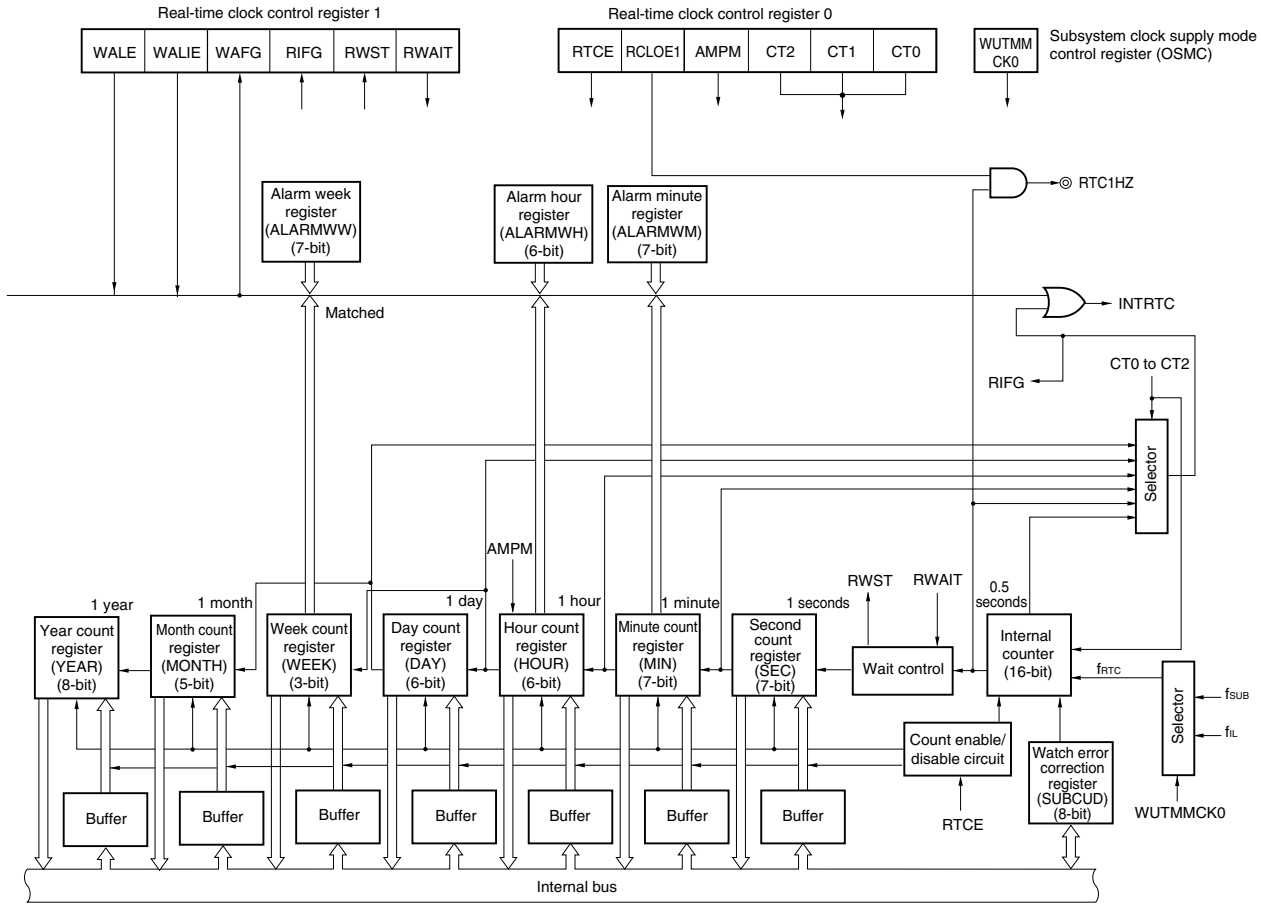
The real-time clock includes the following hardware.

**Table 7-1. Configuration of Real-time Clock**

| Item | Configuration |
|---|---|
| Counter | Internal counter (16-bit) |
| Control registers | Peripheral enable register 0 (PER0) |
| | Subsystem clock supply mode control register (OSMC) |
| | Real-time clock control register 0 (RTCC0) |
| | Real-time clock control register 1 (RTCC1) |
| | Second count register (SEC) |
| | Minute count register (MIN) |
| | Hour count register (HOUR) |
| | Day count register (DAY) |
| | Week count register (WEEK) |
| | Month count register (MONTH) |
| | Year count register (YEAR) |
| | Watch error correction register (SUBCUD) |
| | Alarm minute register (ALARMWM) |
| | Alarm hour register (ALARMWH) |
| | Alarm week register (ALARMWW) |

**Figure 7-1. Block Diagram of Real-time Clock**

<R>




**Caution The count of year, month, week, day, hour, minutes and second can only be performed when a subsystem clock ($f_{SUB}$ = 32.768 kHz) is selected as the operation clock of the real-time clock. When the low-speed oscillation clock ($f_{IL}$ = 15 kHz (TYP.)) is selected, only the constant-period interrupt function is available. The 25- to 32-pin products have the constant-period interrupt function only, because these products have no subsystem clock.**
**However, the constant-period interrupt interval when $f_{IL}$ is selected will be calculated with the constant-period (the value selected with RTCC0 register) × $f_{SUB}/f_{IL}$.**

## 7.3 Registers Controlling Real-time Clock

The real-time clock is controlled by the following registers.

- Peripheral enable register 0 (PER0)
<R> - Subsystem clock supply mode control register (OSMC)
- Real-time clock control register 0 (RTCC0)
- Real-time clock control register 1 (RTCC1)
- Second count register (SEC)
- Minute count register (MIN)
- Hour count register (HOUR)
- Day count register (DAY)
- Week count register (WEEK)
- Month count register (MONTH)
- Year count register (YEAR)
- Watch error correction register (SUBCUD)
- Alarm minute register (ALARMWM)
- Alarm hour register (ALARMWH)
- Alarm week register (ALARMWW)
<R> - Port mode register 3 (PM3)
<R> - Port register 3 (P3)

### 7.3.1 Peripheral enable register 0 (PER0)

This register is used to enable or disable supplying the clock to the peripheral hardware. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

When the real-time clock is used, be sure to set bit 7 (RTCEN) of this register to 1.

The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 7-2. Format of Peripheral Enable Register 0 (PER0)**

Address: F00F0H After reset: 00H R/W

| Symbol | <7> | 6 | <5> | <4> | <3> | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER0 | RTCEN | 0 | ADCEN | IICA0EN | SAU1EN[Note] | SAU0EN | 0 | TAU0EN |

| RTCEN | Control of real-time clock (RTC) and 12-bit interval timer input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFR used by the real-time clock (RTC) and 12-bit interval timer cannot be written.<br>• The real-time clock (RTC) and 12-bit interval timer are in the reset status. |
| 1 | Enables input clock supply.<br>• SFR used by the real-time clock (RTC) and 12-bit interval timer can be read and written. |

**Note** 32, 48, and 64-pin products only.

<R> **Cautions 1. When using the real-time clock, first set the RTCEN bit to 1 and then set the following registers, while oscillation of the count clock ($f_{RTC}$) is stable. If RTCEN = 0, writing to the control registers of the real-time clock is ignored, and, even if the registers are read, only the default values are read (except for the subsystem clock supply mode control register (OSMC), port mode register 3 (PM3), port register 3 (P3)).**

- **Real-time clock control register 0 (RTCC0)**
- **Real-time clock control register 1 (RTCC1)**
- **Second count register (SEC)**
- **Minute count register (MIN)**
- **Hour count register (HOUR)**
- **Day count register (DAY)**
- **Week count register (WEEK)**
- **Month count register (MONTH)**
- **Year count register (YEAR)**
- **Watch error correction register (SUBCUD)**
- **Alarm minute register (ALARMWM)**
- **Alarm hour register (ALARMWH)**
- **Alarm week register (ALARMWW)**

<R> **2. The subsystem clock supply to peripheral functions other than the real-time clock and 12-bit interval timer can be stopped in STOP mode or HALT mode when the subsystem clock is used, by setting the RTCLPC bit of the subsystem clock supply mode control register (OSMC) to 1.**

**3. Be sure to clear the following bits to 0.**

**25-pin products: bits 1, 3, 6**

**32, 48, 64-pin products: bits 1, 6**

### 7.3.2 Subsystem clock supply mode control register (OSMC)

The WUTMMCK0 bit can be used to select the count clock ($f_{RTC}$) of the real-time clock.

In addition, by stopping clock functions that are unnecessary, the RTCLPC bit can be used to reduce power consumption. For details about setting the RTCLPC bit, see **CHAPTER 5 CLOCK GENERATOR**.

The OSMC register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 7-3. Format of Subsystem Clock Supply Mode Control Register (OSMC)**

Address: F00F3H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| OSMC | RTCLPC | 0 | 0 | WUTMMCK0 | 0 | 0 | 0 | 0 |

| WUTMMCK0 | Selection of operation clock ($f_{RTC}$) for real-time clock and 12-bit interval timer. |
|---|---|
| 0 | Subsystem clock ($f_{SUB}$) |
| 1 | Low-speed on-chip oscillator clock ($f_{IL}$) |

**Caution The count of year, month, week, day, hour, minutes and second can only be performed when a subsystem clock ($f_{SUB}$ = 32.768 kHz) is selected as the operation clock of the real-time clock. When the low-speed oscillation clock ($f_{IL}$ = 15 kHz (TYP.)) is selected, only the constant-period interrupt function is available. The 25- to 32-pin products have the constant-period interrupt function only, because these products have no subsystem clock.**
**However, the constant-period interrupt interval when $f_{IL}$ is selected will be calculated with the constant-period (the value selected with RTCC0 register) × $f_{SUB}/f_{IL}$.**

### 7.3.3 Real-time clock control register 0 (RTCC0)

The RTCC0 register is an 8-bit register that is used to start or stop the real-time clock operation, control the RTC1HZ pin, and set a 12- or 24-hour system and the constant-period interrupt function.

The RTCC0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 7-4. Format of Real-time Clock Control Register 0 (RTCC0)**

Address: FFF9DH After reset: 00H R/W

| Symbol | <7> | 6 | <5> | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RTCC0 | RTCE | 0 | RCLOE1[Note] | 0 | AMPM | CT2 | CT1 | CT0 |

| RTCE | Real-time clock operation control |
|---|---|
| 0 | Stops counter operation. |
| 1 | Starts counter operation. |

| RCLOE1 | RTC1HZ pin output control |
|---|---|
| 0 | Disables output of the RTC1HZ pin (1 Hz). |
| 1 | Enables output of the RTC1HZ pin (1 Hz). |

| AMPM | Selection of 12-/24-hour system |
|---|---|
| 0 | 12-hour system (a.m. and p.m. are displayed.) |
| 1 | 24-hour system |
| • Rewrite the AMPM bit value after setting the RWAIT bit (bit 0 of real-time clock control register 1 (RTCC1)) to 1. If the AMPM bit value is changed, the values of the hour count register (HOUR) change according to the specified time system.<br>• Table 7-2 shows the displayed time digits that are displayed. | |

| CT2 | CT1 | CT0 | Constant-period interrupt (INTRTC) selection |
|---|---|---|---|
| 0 | 0 | 0 | Does not use constant-period interrupt function. |
| 0 | 0 | 1 | Once per 0.5 s (synchronized with second count up) |
| 0 | 1 | 0 | Once per 1 s (same time as second count up) |
| 0 | 1 | 1 | Once per 1 m (second 00 of every minute) |
| 1 | 0 | 0 | Once per 1 hour (minute 00 and second 00 of every hour) |
| 1 | 0 | 1 | Once per 1 day (hour 00, minute 00, and second 00 of every day) |
| 1 | 1 | × | Once per 1 month (Day 1, hour 00 a.m., minute 00, and second 00 of every month) |
| When changing the values of the CT2 to CT0 bits while the counter operates (RTCE = 1), rewrite the values of the CT2 to CT0 bits after disabling interrupt servicing INTRTC by using the interrupt mask flag register. Furthermore, after rewriting the values of the CT2 to CT0 bits, enable interrupt servicing after clearing the RIFG and RTCIF flags. | | | |

<R> **Note** Set the RCLOE1 bit to 0 in the 25- and 32-pin products.

**Cautions 1. Do not change the value of the RCLOE1 bit when RTCE = 1.**

<R> **2. 1 Hz is not output even if RCCOE1 is set to 1 when RTCE = 0.**

**Remark** ×: don't care

### 7.3.4 Real-time clock control register 1 (RTCC1)

The RTCC1 register is an 8-bit register that is used to control the alarm interrupt function and the wait time of the counter.

The RTCC1 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 7-5. Format of Real-time Clock Control Register 1 (RTCC1) (1/2)**

Address: FFF9EH After reset: 00H R/W

| Symbol | <7> | <6> | 5 | <4> | <3> | 2 | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| RTCC1 | WALE | WALIE | 0 | WAFG | RIFG | 0 | RWST | RWAIT |

| WALE | Alarm operation control |
|---|---|
| 0 | Match operation is invalid. |
| 1 | Match operation is valid. |
| When setting a value to the WALE bit while the counter operates (RTCE = 1) and WALIE = 1, rewrite the WALE bit after disabling interrupt servicing INTRTC by using the interrupt mask flag register. Furthermore, clear the WAFG and RTCIF flags after rewriting the WALE bit. When setting each alarm register (WALIE flag of real-time clock control register 1 (RTCC1), the alarm minute register (ALARMWM), the alarm hour register (ALARMWH), and the alarm week register (ALARMWW)), set match operation to be invalid (“0”) for the WALE bit. | |

| WALIE | Control of alarm interrupt (INTRTC) function operation |
|---|---|
| 0 | Does not generate interrupt on matching of alarm. |
| 1 | Generates interrupt on matching of alarm. |

| WAFG | Alarm detection status flag |
|---|---|
| 0 | Alarm mismatch |
| 1 | Detection of matching of alarm |
| This is a status flag that indicates detection of matching with the alarm. It is valid only when WALE = 1 and is set to “1” one cycle of $f_{RTC}$ after matching of the alarm is detected. This flag is cleared when “0” is written to it. Writing “1” to it is invalid. | |

<R>

| RIFG | Constant-period interrupt status flag |
|---|---|
| 0 | Constant-period interrupt is not generated. |
| 1 | Constant-period interrupt is generated. |
| This flag indicates the status of generation of the constant-period interrupt. When the constant-period interrupt is generated, it is set to “1”.<br>This flag is cleared when “0” is written to it. Writing “1” to it is invalid. | |

**Figure 7-5. Format of Real-time Clock Control Register 1 (RTCC1) (2/2)**

Address: FFF9EH After reset: 00H R/W

| Symbol | <7> | <6> | 5 | <4> | <3> | 2 | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| RTCC1 | WALE | WALIE | 0 | WAFG | RIFG | 0 | RWST | RWAIT |

| RWST | Wait status flag of real-time clock |
|---|---|
| 0 | Counter is operating. |
| 1 | Mode to read or write counter value |
| This status flag indicates whether the setting of the RWAIT bit is valid.<br>Before reading or writing the counter value, confirm that the value of this flag is 1. | |

| RWAIT | Wait control of real-time clock |
|---|---|
| 0 | Sets counter operation. |
| 1 | Stops SEC to YEAR counters. Mode to read or write counter value |
| This bit controls the operation of the counter.<br>Be sure to write “1” to it to read or write the counter value.<br>As the internal counter (16-bit) is continuing to run, complete reading or writing within one second and turn back to 0.<br>When RWAIT = 1, it takes up to one cycle of $f_{RTC}$ until the counter value can be read or written (RWST = 1).<br>When the internal counter (16-bit) overflowed while RWAIT = 1, it keeps the event of overflow until RWAIT = 0, then counts up.<br>However, when it wrote a value to second count register, it will not keep the overflow event. | |

**Caution If writing is performed to the RTCC1 register with a 1-bit manipulation instruction, the RIFG flag and WAFG flag may be cleared. Therefore, to perform writing to the RTCC1 register, be sure to use an 8-bit manipulation instruction. To prevent the RIFG flag and WAFG flag from being cleared during writing, disable writing by setting 1 to the corresponding bit. If the RIFG flag and WAFG flag are not used and the value may be changed, the RTCC1 register may be written by using a 1-bit manipulation instruction.**

**Remarks 1.** Fixed-cycle interrupts and alarm match interrupts use the same interrupt source (INTRTC). When using these two types of interrupts at the same time, which interrupt occurred can be judged by checking the fixed-cycle interrupt status flag (RIFG) and the alarm detection status flag (WAFG) upon INTRTC occurrence.

<R> **2.** The internal counter (16 bits) is cleared when the second count register (SEC) is written.

### 7.3.5 Second count register (SEC)

The SEC register is an 8-bit register that takes a value of 0 to 59 (decimal) and indicates the count value of seconds.

It counts up when the internal counter (16-bit) overflows.

<R> When data is written to this register, it is written to a buffer and then to the counter up to two cycles of $f_{RTC}$ later. Set a decimal value of 00 to 59 to this register in BCD code.

The SEC register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

<R> **Remark** The internal counter (16 bits) is cleared when the second count register (SEC) is written.

**Figure 7-6. Format of Second Count Register (SEC)**

Address: FFF92H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| SEC | 0 | SEC40 | SEC20 | SEC10 | SEC8 | SEC4 | SEC2 | SEC1 |

**Caution When it reads or writes from/to the register while the counter is in operation (RTCE = 1), follow the procedures described in 7.4.3 Reading/writing real-time clock.**

### 7.3.6 Minute count register (MIN)

The MIN register is an 8-bit register that takes a value of 0 to 59 (decimal) and indicates the count value of minutes.

It counts up when the second counter overflows.

<R> When data is written to this register, it is written to a buffer and then to the counter up to two cycles of $f_{RTC}$ later. Even if the second count register overflows while this register is being written, this register ignores the overflow and is set to the value written. Set a decimal value of 00 to 59 to this register in BCD code.

The MIN register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 7-7. Format of Minute Count Register (MIN)**

Address: FFF93H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| MIN | 0 | MIN40 | MIN20 | MIN10 | MIN8 | MIN4 | MIN2 | MIN1 |

**Caution When it reads or writes from/to the register while the counter is in operation (RTCE = 1), follow the procedures described in 7.4.3 Reading/writing real-time clock.**

### 7.3.7 Hour count register (HOUR)

The HOUR register is an 8-bit register that takes a value of 00 to 23 or 01 to 12 and 21 to 32 (decimal) and indicates the count value of hours.

It counts up when the minute counter overflows.

<R> When data is written to this register, it is written to a buffer and then to the counter up to two cycles of $f_{RTC}$ later. Even if the minute count register overflows while this register is being written, this register ignores the overflow and is set to the value written. Specify a decimal value of 00 to 23, 01 to 12, or 21 to 32 by using BCD code according to the time system specified using bit 3 (AMPM) of real-time clock control register 0 (RTCC0).

If the AMPM bit value is changed, the values of the HOUR register change according to the specified time system.

The HOUR register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 12H.

However, the value of this register is 00H if the AMPM bit is set to 1 after reset.

**Figure 7-8. Format of Hour Count Register (HOUR)**

Address: FFF94H After reset: 12H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| HOUR | 0 | 0 | HOUR20 | HOUR10 | HOUR8 | HOUR4 | HOUR2 | HOUR1 |

**Cautions 1. Bit 5 (HOUR20) of the HOUR register indicates AM(0)/PM(1) if AMPM = 0 (if the 12-hour system is selected).**

**2. When it reads or writes from/to the register while the counter is in operation (RTCE = 1), follow the procedures described in 7.4.3 Reading/writing real-time clock.**

Table 7-2 shows the relationship between the setting value of the AMPM bit, the hour count register (HOUR) value, and time.

**Table 7-2. Displayed Time Digits**

| 24-Hour Display (AMPM = 1) | | 12-Hour Display (AMPM = 1) | |
|---|---|---|---|
| Time | HOUR Register | Time | HOUR Register |
| 0 | 00H | 12 a.m. | 12H |
| 1 | 01H | 1 a.m. | 01H |
| 2 | 02H | 2 a.m. | 02H |
| 3 | 03H | 3 a.m. | 03H |
| 4 | 04H | 4 a.m. | 04H |
| 5 | 05H | 5 a.m. | 05H |
| 6 | 06H | 6 a.m. | 06H |
| 7 | 07H | 7 a.m. | 07H |
| 8 | 08H | 8 a.m. | 08H |
| 9 | 09H | 9 a.m. | 09H |
| 10 | 10H | 10 a.m. | 10H |
| 11 | 11H | 11 a.m. | 11H |
| 12 | 12H | 12 p.m. | 32H |
| 13 | 13H | 1 p.m. | 21H |
| 14 | 14H | 2 p.m. | 22H |
| 15 | 15H | 3 p.m. | 23H |
| 16 | 16H | 4 p.m. | 24H |
| 17 | 17H | 5 p.m. | 25H |
| 18 | 18H | 6 p.m. | 26H |
| 19 | 19H | 7 p.m. | 27H |
| 20 | 20H | 8 p.m. | 28H |
| 21 | 21H | 9 p.m. | 29H |
| 22 | 22H | 10 p.m. | 30H |
| 23 | 23H | 11 p.m. | 31H |

The HOUR register value is set to 12-hour display when the AMPM bit is “0” and to 24-hour display when the AMPM bit is “1”.

In 12-hour display, the fifth bit of the HOUR register displays 0 for AM and 1 for PM.

### 7.3.8 Day count register (DAY)

The DAY register is an 8-bit register that takes a value of 1 to 31 (decimal) and indicates the count value of days.

It counts up when the hour counter overflows.

This counter counts as follows.

- 01 to 31 (January, March, May, July, August, October, December)
- 01 to 30 (April, June, September, November)
- 01 to 29 (February, leap year)
- 01 to 28 (February, normal year)

<R> When data is written to this register, it is written to a buffer and then to the counter up to two cycles of $f_{RTC}$ later. Even if the hour count register overflows while this register is being written, this register ignores the overflow and is set to the value written. Set a decimal value of 01 to 31 to this register in BCD code.

The DAY register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 01H.

**Figure 7-9. Format of Day Count Register (DAY)**

Address: FFF96H After reset: 01H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DAY | 0 | 0 | DAY20 | DAY10 | DAY8 | DAY4 | DAY2 | DAY1 |

**Caution When it reads or writes from/to the register while the counter is in operation (RTCE = 1), follow the procedures described in 7.4.3 Reading/writing real-time clock.**

### 7.3.9 Week count register (WEEK)

The WEEK register is an 8-bit register that takes a value of 0 to 6 (decimal) and indicates the count value of weekdays.

It counts up in synchronization with the day counter.

<R> When data is written to this register, it is written to a buffer and then to the counter up to two cycles of $f_{RTC}$ later. Set a decimal value of 00 to 06 to this register in BCD code.

The WEEK register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 7-10. Format of Week Count Register (WEEK)**

Address: FFF95H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| WEEK | 0 | 0 | 0 | 0 | 0 | WEEK4 | WEEK2 | WEEK1 |

**Cautions 1. The value corresponding to the month count register (MONTH) or the day count register (DAY) is not stored in the week count register (WEEK) automatically. After reset release, set the week count register as follow.**

| Day | WEEK |
|---|---|
| Sunday | 00H |
| Monday | 01H |
| Tuesday | 02H |
| Wednesday | 03H |
| Thursday | 04H |
| Friday | 05H |
| Saturday | 06H |

**2. When it reads or writes from/to the register while the counter is in operation (RTCE = 1), follow the procedures described in 7.4.3 Reading/writing real-time clock.**

### 7.3.10 Month count register (MONTH)

The MONTH register is an 8-bit register that takes a value of 1 to 12 (decimal) and indicates the count value of months.

It counts up when the day counter overflows.

<R> When data is written to this register, it is written to a buffer and then to the counter up to two cycles of $f_{RTC}$ later. Even if the day count register overflows while this register is being written, this register ignores the overflow and is set to the value written. Set a decimal value of 01 to 12 to this register in BCD code.

The MONTH register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 01H.

**Figure 7-11. Format of Month Count Register (MONTH)**

Address: FFF97H After reset: 01H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| MONTH | 0 | 0 | 0 | MONTH10 | MONTH8 | MONTH4 | MONTH2 | MONTH1 |

**Caution When it reads or writes from/to the register while the counter is in operation (RTCE = 1), follow the procedures described in 7.4.3 Reading/writing real-time clock.**

### 7.3.11 Year count register (YEAR)

The YEAR register is an 8-bit register that takes a value of 0 to 99 (decimal) and indicates the count value of years.

It counts up when the month count register (MONTH) overflows.

Values 00, 04, 08, …, 92, and 96 indicate a leap year.

<R> When data is written to this register, it is written to a buffer and then to the counter up to two cycles of $f_{RTC}$ later. Even if the MONTH register overflows while this register is being written, this register ignores the overflow and is set to the value written. Set a decimal value of 00 to 99 to this register in BCD code.

The YEAR register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 7-12. Format of Year Count Register (YEAR)**

Address: FFF98H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| YEAR | YEAR80 | YEAR40 | YEAR20 | YEAR10 | YEAR8 | YEAR4 | YEAR2 | YEAR1 |

**Caution When it reads or writes from/to the register while the counter is in operation (RTCE = 1), follow the procedures described in 7.4.3 Reading/writing real-time clock.**

### 7.3.12 Watch error correction register (SUBCUD)

This register is used to correct the watch with high accuracy when it is slow or fast by changing the value that overflows from the internal counter (16-bit) to the second count register (SEC) (reference value: 7FFFH).

The SUBCUD register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 7-13. Format of Watch Error Correction Register (SUBCUD)**

Address: FFF99H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| SUBCUD | DEV | F6 | F5 | F4 | F3 | F2 | F1 | F0 |

| DEV | Setting of watch error correction timing |
|---|---|
| 0 | Corrects watch error when the second digits are at 00, 20, or 40 (every 20 seconds). |
| 1 | Corrects watch error only when the second digits are at 00 (every 60 seconds). |
| Writing to the SUBCUD register at the following timing is prohibited.<br>• When DEV = 0 is set: For a period of SEC = 00H, 20H, 40H<br>• When DEV = 1 is set: For a period of SEC = 00H | |

| F6 | Setting of watch error correction value |
|---|---|
| 0 | Increases by {(F5, F4, F3, F2, F1, F0) – 1} × 2. |
| 1 | Decreases by {(/F5, /F4, /F3, /F2, /F1, /F0) + 1} × 2. |
| When (F6, F5, F4, F3, F2, F1, F0) = (*, 0, 0, 0, 0, 0, *), the watch error is not corrected. * is 0 or 1.<br>/F5 to /F0 are the inverted values of the corresponding bits (000011 when 111100).<br>Range of correction value: (when F6 = 0) 2, 4, 6, 8, … , 120, 122, 124<br>(when F6 = 1) –2, –4, –6, –8, … , –120, –122, –124 | |

The range of value that can be corrected by using the watch error correction register (SUBCUD) is shown below.

| | DEV = 0 (correction every 20 seconds) | DEV = 1 (correction every 60 seconds) |
|---|---|---|
| Correctable range | –189.2 ppm to 189.2 ppm | –63.1 ppm to 63.1 ppm |
| Maximum excludes quantization error | ± 1.53 ppm | ± 0.51 ppm |
| Minimum resolution | ± 3.05 ppm | ± 1.02 ppm |

**Remark** If a correctable range is –63.1 ppm or lower and 63.1 ppm or higher, set 0 to DEV.

### 7.3.13 Alarm minute register (ALARMWM)

This register is used to set minutes of alarm.

The ALARMWM register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Caution Set a decimal value of 00 to 59 to this register in BCD code. If a value outside the range is set, the alarm is not detected.**

**Figure 7-14. Format of Alarm Minute Register (ALARMWM)**

Address: FFF9AH After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ALARMWM | 0 | WM40 | WM20 | WM10 | WM8 | WM4 | WM2 | WM1 |

### 7.3.14 Alarm hour register (ALARMWH)

This register is used to set hours of alarm.

The ALARMWH register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 12H.

However, the value of this register is 00H if the AMPM bit is set to 1 after reset.

**Caution Set a decimal value of 00 to 23, 01 to 12, or 21 to 32 to this register in BCD code. If a value outside the range is set, the alarm is not detected.**

**Figure 7-15. Format of Alarm Hour Register (ALARMWH)**

Address: FFF9BH After reset: 12H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ALARMWH | 0 | 0 | WH20 | WH10 | WH8 | WH4 | WH2 | WH1 |

**Caution Bit 5 (WH20) of the ALARMWH register indicates AM(0)/PM(1) if AMPM = 0 (if the 12-hour system is selected).**

### 7.3.15 Alarm week register (ALARMWW)

This register is used to set date of alarm.

The ALARMWW register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 7-16. Format of Alarm Week Register (ALARMWW)**

Address: FFF9CH After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ALARMWW | 0 | WW6 | WW5 | WW4 | WW3 | WW2 | WW1 | WW0 |

Here is an example of setting the alarm.

| Time of Alarm | Day | | | | | | | 12-Hour Display | | | | 24-Hour Display | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Sunday<br>WW0 | Monday<br>WW1 | Tuesday<br>WW2 | Wednesday<br>WW3 | Thursday<br>WW4 | Friday<br>WW5 | Saturday<br>WW6 | Hour 10 | Hour 1 | Minute 10 | Minute 1 | Hour 10 | Hour 1 | Minute 10 | Minute 1 |
| Every day, 0:00 a.m. | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 0 | 0 | 0 | 0 | 0 | 0 |
| Every day, 1:30 a.m. | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 3 | 0 | 0 | 1 | 3 | 0 |
| Every day, 11:59 a.m. | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 5 | 9 | 1 | 1 | 5 | 9 |
| Monday through Friday, 0:00 p.m. | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 3 | 2 | 0 | 0 | 1 | 2 | 0 | 0 |
| Sunday, 1:30 p.m. | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 1 | 3 | 0 | 1 | 3 | 3 | 0 |
| Monday, Wednesday, Friday, 11:59 p.m. | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 3 | 1 | 5 | 9 | 2 | 3 | 5 | 9 |

### <R> 7.3.16 Port mode register 3 (PM3)

The PM3 register can be set by a 1-bit or 8-bit manipulation instruction.

Reset signal generation sets this register to FFH.

When using the port 3 as the RTC1HZ pin for output of 1 Hz, set the PM30 bit to 0.

**Figure 7-17. Format of Port Mode Register 3 (PM3)**

Address: FFF23H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM3 | 1 | 1 | 1 | 1 | 1 | 1 | PM31 | PM30 |

### <R> 7.3.17 Port register 3 (P3)

The P3 register can be set by a 1-bit or 8-bit manipulation instruction.

Reset signal generation sets this register to 00H.

When using the port 3 as 1-Hz output to the RTC1Hz pin, set the P30 bit to 0.

**Figure 7-18. Format of Port Register 3 (P3)**

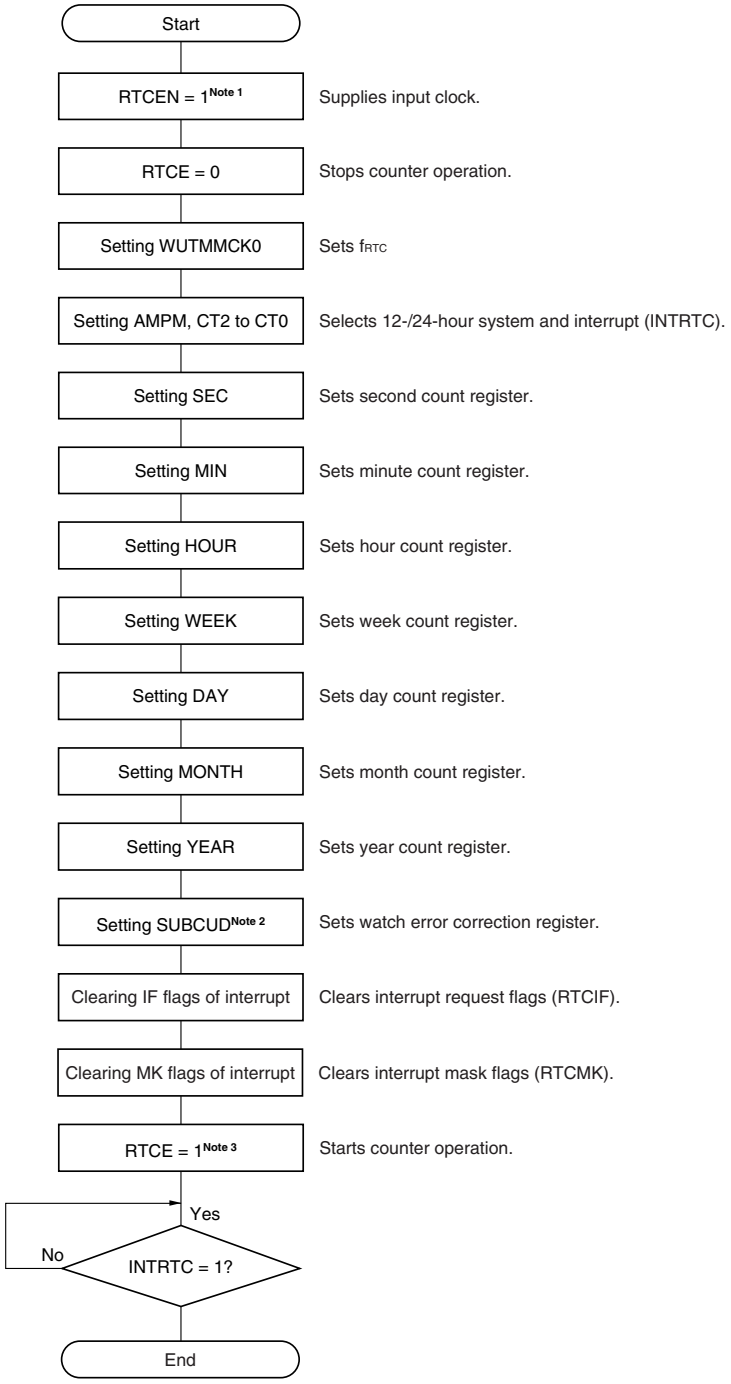
Address: FFF03H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| P3 | 0 | 0 | 0 | 0 | 0 | 0 | P31 | P30 |

## 7.4 Real-time Clock Operation

### 7.4.1 Starting operation of real-time clock

**Figure 7-19. Procedure for Starting Operation of Real-time Clock**




**Notes 1.** First set the RTCEN bit to 1, while oscillation of the count clock ($f_{RTC}$) is stable.

**2.** Set up the SUBCUD register only if the watch error must be corrected. For details about how to calculate the correction value, see **7.4.6 Example of watch error correction of real-time clock**.

**3.** Confirm the procedure described in **7.4.2 Shifting to HALT/STOP mode after starting operation** when shifting to HALT/STOP mode without waiting for INTRTC = 1 after RTCE = 1.

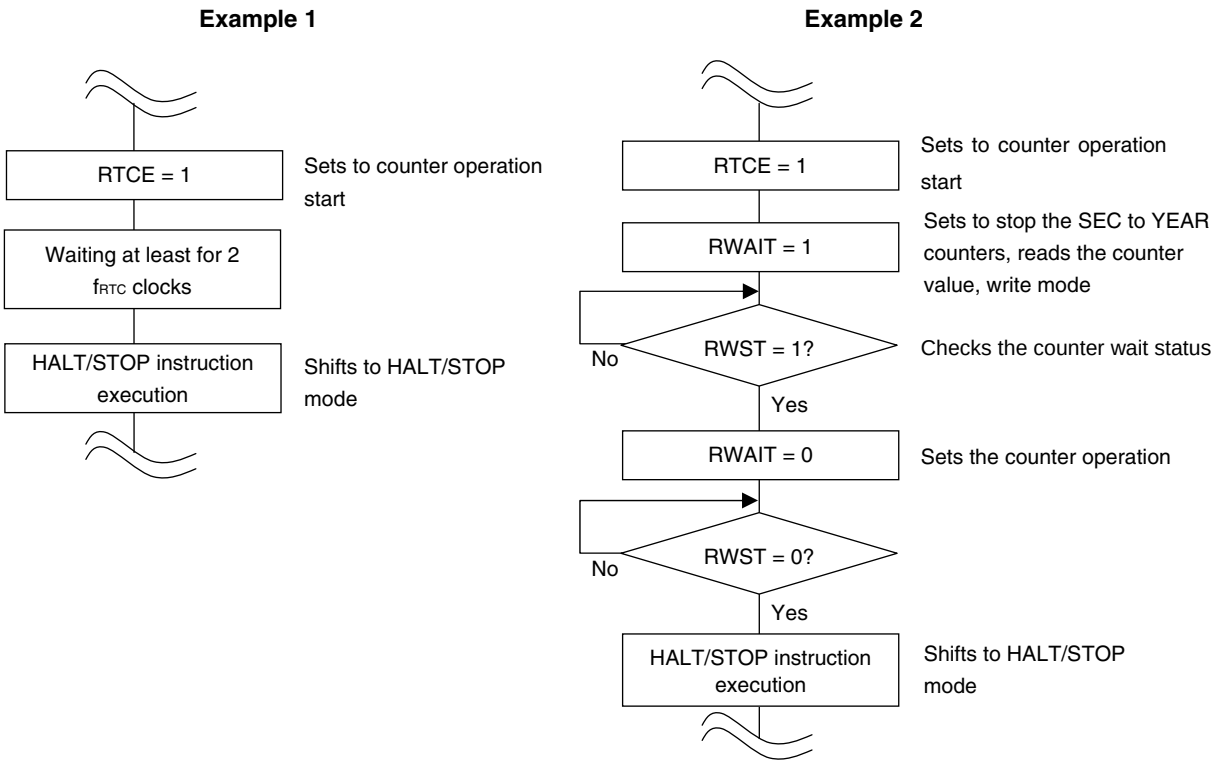
### 7.4.2 Shifting to HALT/STOP mode after starting operation

Perform one of the following processing when shifting to HALT/STOP mode immediately after setting the RTCE bit to 1.

However, after setting the RTCE bit to 1, this processing is not required when shifting to HALT/STOP mode after the INTRTC interrupt has occurred.

- Shifting to HALT/STOP mode when at least two count clocks ($f_{RTC}$) have elapsed after setting the RTCE bit to 1 (see **Figure 7-20**, **Example 1**).
- Checking by polling the RWST bit to become 1, after setting the RTCE bit to 1 and then setting the RWAIT bit to 1. Afterward, setting the RWAIT bit to 0 and shifting to HALT/STOP mode after checking again by polling that the RWST bit has become 0 (see **Figure 7-20**, **Example 2**).

**Figure 7-20. Procedure for Shifting to HALT/STOP Mode After Setting RTCE bit to 1**

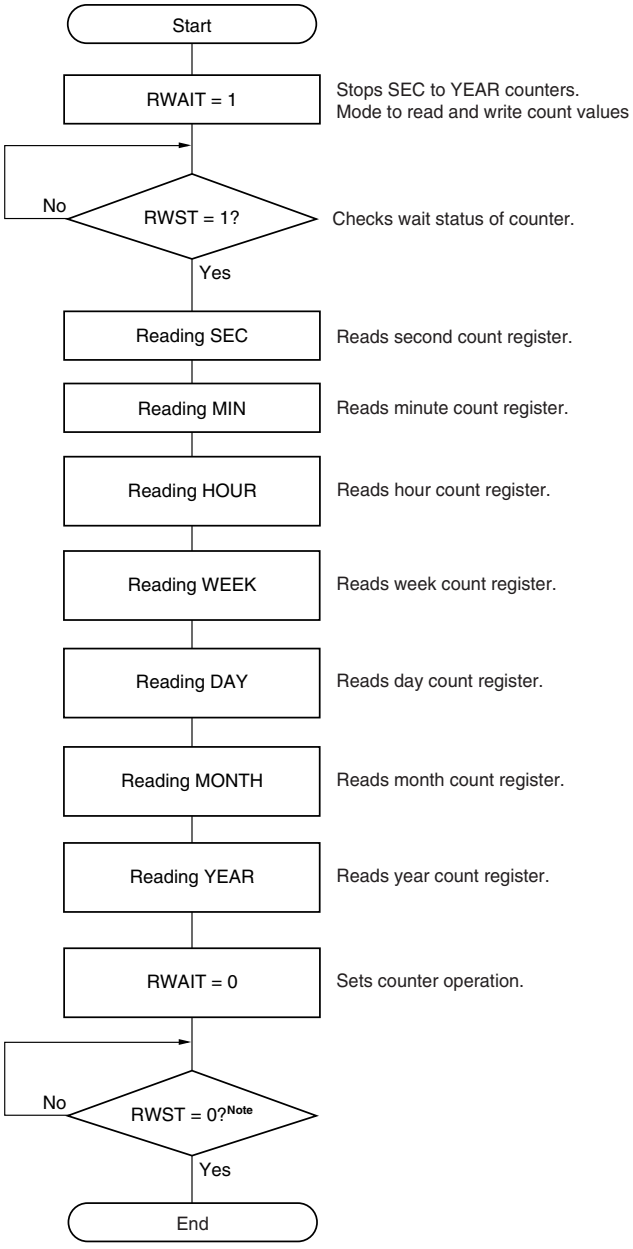


### 7.4.3 Reading/writing real-time clock

Read or write the counter after setting 1 to RWAIT first.

Set RWAIT to 0 after completion of reading or writing the counter.

**Figure 7-21. Procedure for Reading Real-time Clock**



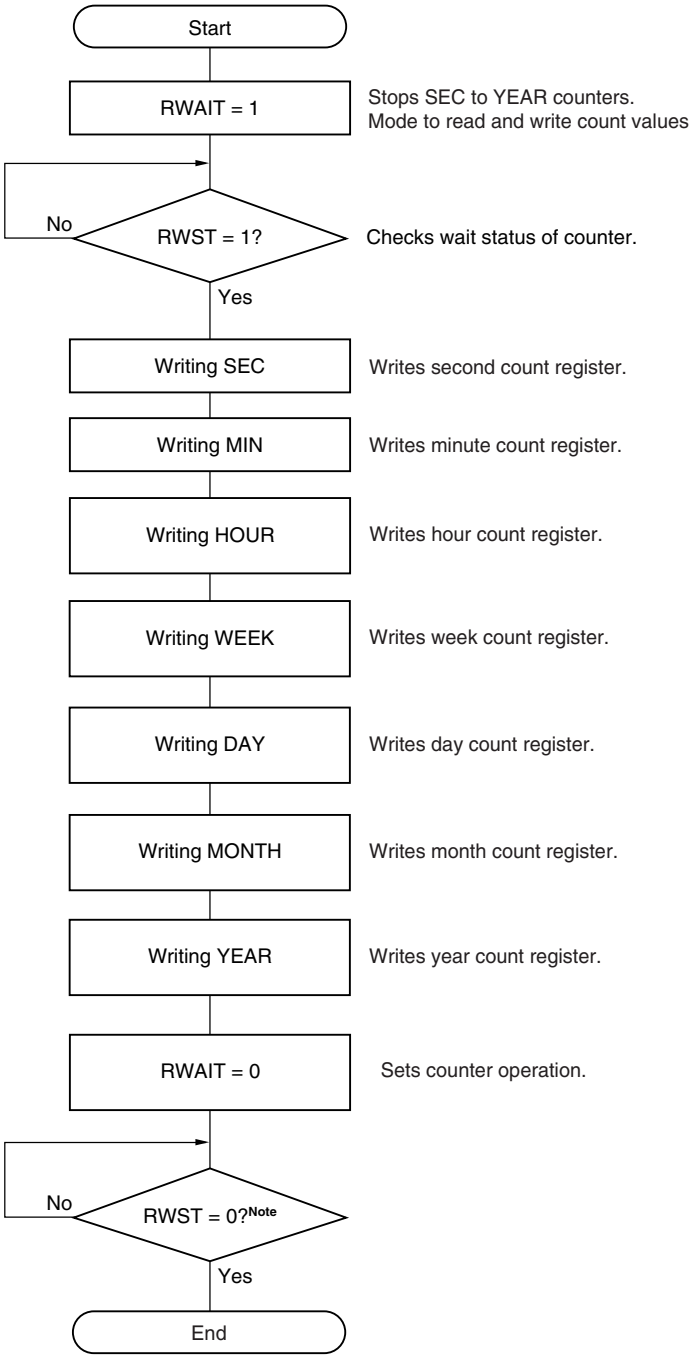

**Note** Be sure to confirm that RWST = 0 before setting STOP mode.

**Caution Complete the series of process of setting the RWAIT bit to 1 to clearing the RWAIT bit to 0 within 1 second.**

**Remark** The second count register (SEC), minute count register (MIN), hour count register (HOUR), week count register (WEEK), day count register (DAY), month count register (MONTH), and year count register (YEAR) may be read in any sequence.

All the registers do not have to read and only some registers may be read.

**Figure 7-22. Procedure for Writing Real-time Clock**




**Note** Be sure to confirm that RWST = 0 before setting STOP mode.

**Cautions 1. Complete the series of operations of setting the RWAIT bit to 1 to clearing the RWAIT bit to 0 within 1 second.**

**2. When changing the values of the SEC, MIN, HOUR, WEEK, DAY, MONTH, and YEAR register while the counter operates (RTCE = 1), rewrite the values of the MIN register after disabling interrupt servicing INTRTC by using the interrupt mask flag register. Furthermore, clear the WAFG, RIFG and RTCIF flags after rewriting the MIN register.**

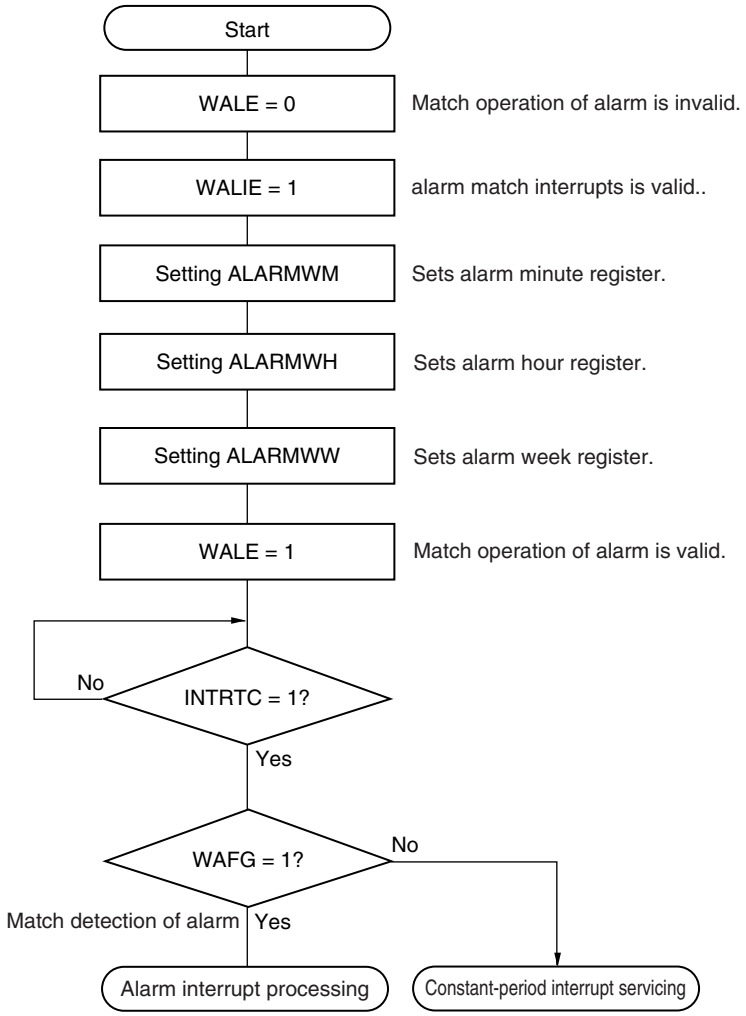
**Remark** The second count register (SEC), minute count register (MIN), hour count register (HOUR), week count register (WEEK), day count register (DAY), month count register (MONTH), and year count register (YEAR) may be written in any sequence.

All the registers do not have to be set and only some registers may be written.

### 7.4.4 Setting alarm of real-time clock

Set time of alarm after setting 0 to WALE (alarm operation invalid.) first.

**Figure 7-23. Alarm processing Procedure**

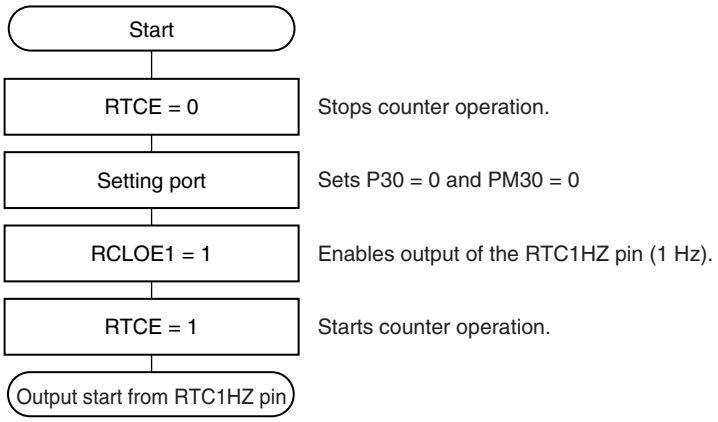



**Remarks 1.** The alarm week register (ALARMWW), alarm hour register (ALARMWH), and alarm week register (ALARMWW) may be written in any sequence.

**2.** Fixed-cycle interrupts and alarm match interrupts use the same interrupt source (INTRTC). When using these two types of interrupts at the same time, which interrupt occurred can be judged by checking the fixed-cycle interrupt status flag (RIFG) and the alarm detection status flag (WAFG) upon INTRTC occurrence.

### 7.4.5 1 Hz output of real-time clock

**Figure 7-24. 1 Hz Output Setting Procedure**




<R>

**Cautions 1. First set the RTCEN bit to 1, while oscillation of the count clock ($f_{SUB}$) is stable.**

<R> **2. 1 Hz output function of real-time clock is not available in the 25- and 32-pin products.**

### 7.4.6 Example of watch error correction of real-time clock

The watch can be corrected with high accuracy when it is slow or fast, by setting a value to the watch error correction register.

Example of calculating the correction value

The correction value used when correcting the count value of the internal counter (16-bit) is calculated by using the following expression.

Set the DEV bit to 0 when the correction range is –63.1 ppm or less, or 63.1 ppm or more.

(When DEV = 0)

Correction value[Note] = Number of correction counts in 1 minute ÷ 3 = (Oscillation frequency ÷ Target frequency – 1) × 32768 × 60 ÷ 3

(When DEV = 1)

Correction value[Note] = Number of correction counts in 1 minute = (Oscillation frequency ÷ Target frequency – 1) × 32768 × 60

**Note** The correction value is the watch error correction value calculated by using bits 6 to 0 of the watch error correction register (SUBCUD).

(When F6 = 0) Correction value = {(F5, F4, F3, F2, F1, F0) – 1} × 2

(When F6 = 1) Correction value = – {(/F5, /F4, /F3, /F2, /F1, /F0) + 1} × 2

When (F6, F5, F4, F3, F2, F1, F0) is (*, 0, 0, 0, 0, 0, *), watch error correction is not performed. "*" is 0 or 1. /F5 to /F0 are bit-inverted values (000011 when 111100).

**Remarks 1.** The correction value is 2, 4, 6, 8, … 120, 122, 124 or –2, –4, –6, –8, … –120, –122, –124.

**2.** The oscillation frequency is a value of the count clock ($f_{RTC}$).
It can be calculated from the output frequency of the RTC1HZ pin × 32768 when the watch error correction register is set to its initial value (00H).

**3.** The target frequency is the frequency resulting after correction performed by using the watch error correction register.

<R> Correction example 1

Example of correcting from 32772.3 Hz to 32768 Hz (32772.3 Hz – 131.2 ppm)

[Measuring the oscillation frequency]

The oscillation frequency[Note] of each product is measured by outputting about 32.768 kHz from the PCLBUZ0 pin, or by outputting about 1 Hz from the RTC1HZ pin when the watch error correction register (SUBCUD) is set to its initial value (00H).

**Note** See **7.4.5 1 Hz output of real-time clock** for the setting procedure of the RTC1Hz output, and see **9.4 Operations of Clock Output/Buzzer Output Controller** for the setting procedure of outputting about 32 kHz from the PCLBUZ0 pin.

[Calculating the correction value]

(When the output frequency from the PCLBUZ0 pin is 32772.3 Hz)

Assume the target frequency to be 32768 Hz (32772.3 Hz–131.2 ppm) and DEV to be 0, because the correctable range of –131.2 ppm is –63.1 ppm or lower.

The expression for calculating the correction value when DEV is 0 is applied.

Correction value = Number of correction counts in 1 minute ÷ 3
= (Oscillation frequency ÷ target frequency – 1) × 32768 × 60 ÷ 3
= (32772.3 ÷ 32768 – 1) × 32768 × 60 ÷ 3
= 86

[Calculating the values to be set to (F6 to F0)]

(When the correction value is 86)

If the correction value is 0 or larger (when slowing), assume F6 to be 0.

Calculate (F5, F4, F3, F2, F1, F0) from the correction value.

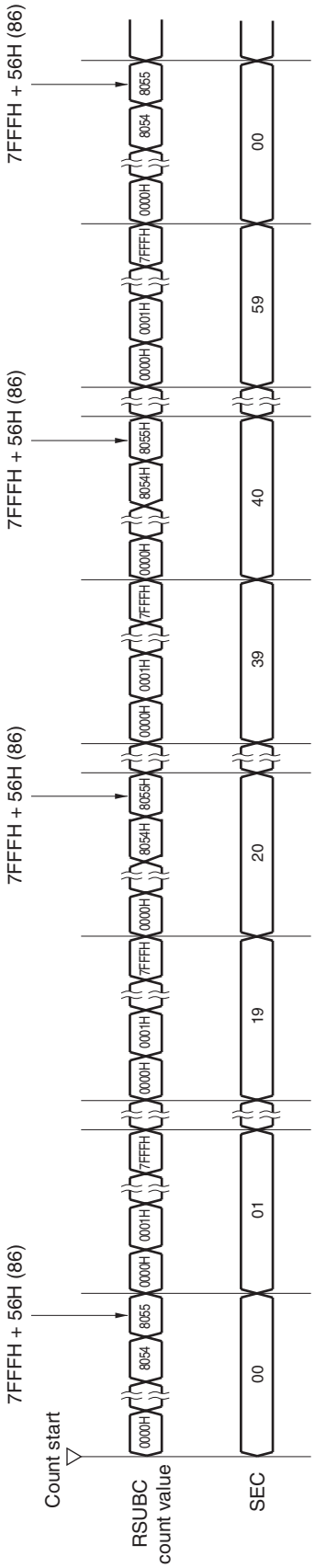
{ (F5, F4, F3, F2, F1, F0) – 1} × 2 = 86
(F5, F4, F3, F2, F1, F0) = 44
(F5, F4, F3, F2, F1, F0) = (1, 0, 1, 1, 0, 0)

Consequently, when correcting from 32772.3 Hz to 32768 Hz (32772.3 Hz – 131.2 ppm), setting the correction register such that DEV is 0 and the correction value is 86 (bits 6 to 0 of the SUBCUD register: 0101100) results in 32768 Hz (0 ppm).

Figure 7-25 shows the operation when (DEV, F6, F5, F4, F3, F2, F1, F0) is (0, 0, 1, 0, 1, 1, 0, 0).

<R>

**Figure 7-25. Correction Operation when (DEV, F6, F5, F4, F3, F2, F1, F0) = (0, 0, 1, 0, 1, 1, 0, 0)**



Correction example 2

Example of correcting from 32767.4 Hz to 32768 Hz (32767.4 Hz + 18.3 ppm)

[Measuring the oscillation frequency]

The oscillation frequency[Note] of each product is measured by outputting about 1 Hz from the RTC1HZ pin when the watch error correction register (SUBCUD) is set to its initial value (00H).

**Note** See **7.4.5 1 Hz output of real-time clock** for the setting procedure of outputting about 1 Hz from the RTC1HZ pin.

[Calculating the correction value]

(When the output frequency from the RTCCL pin is 0.9999817 Hz)

Oscillation frequency = 32768 × 0.9999817 ≈ 32767.4 Hz

Assume the target frequency to be 32768 Hz (32767.4 Hz + 18.3 ppm) and DEV to be 1.

The expression for calculating the correction value when DEV is 1 is applied.

Correction value = Number of correction counts in 1 minute
= (Oscillation frequency ÷ Target frequency − 1) × 32768 × 60
= (32767.4 ÷ 32768 − 1) × 32768 × 60
= −36

[Calculating the values to be set to (F6 to F0)]

(When the correction value is −36)

If the correction value is 0 or less (when quickening), assume F6 to be 1.

Calculate (F5, F4, F3, F2, F1, F0) from the correction value.

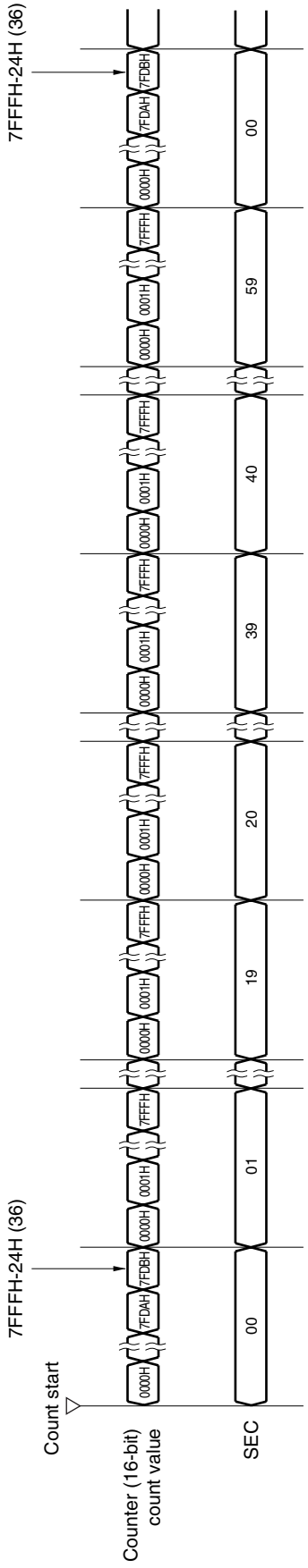
| | |
|---|---|
| − {(/F5, /F4, /F3, /F2, /F1, /F0) − 1} × 2 | = −36 |
| (/F5, /F4, /F3, /F2, /F1, /F0) | = 17 |
| (/F5, /F4, /F3, /F2, /F1, /F0) | = (0, 1, 0, 0, 0, 1) |
| (F5, F4, F3, F2, F1, F0) | = (1, 0, 1, 1, 1, 0) |

Consequently, when correcting from 32767.4 Hz to 32768 Hz (32767.4 Hz + 18.3 ppm), setting the correction register such that DEV is 1 and the correction value is −36 (bits 6 to 0 of the SUBCUD register: 1101110) results in 32768 Hz (0 ppm).

Figure 7-26 shows the operation when (DEV, F6, F5, F4, F3, F2, F1, F0) is (1, 1, 1, 0, 1, 1, 1, 0).

<R>

**Figure 7-26. Correction Operation when (DEV, F6, F5, F4, F3, F2, F1, F0) = (1, 1, 1, 0, 1, 1, 1, 0)**



# CHAPTER 8 12-BIT INTERVAL TIMER

## 8.1 Functions of 12-bit Interval Timer

An interrupt (INTIT) is generated at any previously specified time interval. It can be utilized for wakeup from STOP mode and triggering an A/D converter's SNOOZE mode.

## 8.2 Configuration of 12-bit Interval Timer

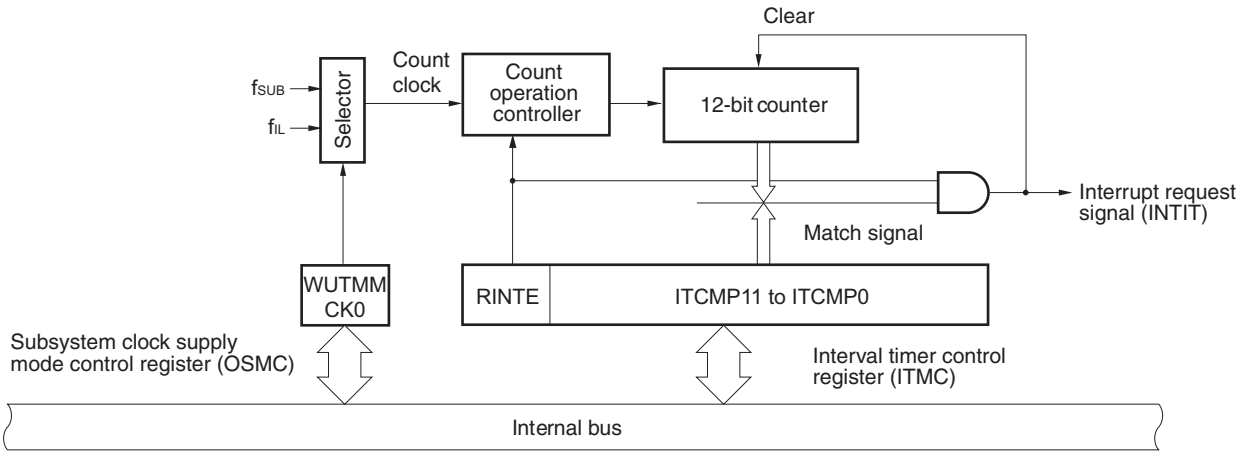
The interval timer includes the following hardware.

**Table 8-1. Configuration of 12-bit Interval Timer**

| Item | Configuration |
|---|---|
| Counter | 12-bit counter |
| Control registers | Peripheral enable register 0 (PER0) |
| | Subsystem clock supply mode control register (OSMC) |
| | Interval timer control register (ITMC) |

**Figure 8-1. Block Diagram of 12-bit Interval Timer**

<R>




<R> **Caution The subsystem clock ($f_{SUB}$) is selectable as a count clock in the 48- and 64-pin products.**

## 8.3 Registers Controlling 12-bit Interval Timer

The 12-bit interval timer is controlled by the following registers.

- Peripheral enable register 0 (PER0)
- Subsystem clock supply mode control register (OSMC)
- Interval timer control register (ITMC)

### 8.3.1 Peripheral enable register 0 (PER0)

This register is used to enable or disable supplying the clock to the peripheral hardware. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

When the 12-bit interval timer is used, be sure to set bit 7 (RTCEN) of this register to 1.

The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 8-2. Format of Peripheral Enable Register 0 (PER0)**

Address: F00F0H After reset: 00H R/W

| Symbol | <7> | 6 | <5> | <4> | <3> | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER0 | RTCEN | 0 | ADCEN | IICA0EN | SAU1EN[Note] | SAU0EN | 0 | TAU0EN |

| RTCEN | Control of real-time clock (RTC) and 12-bit interval timer input clock supply |
|---|---|
| 0 | Stops clock supply.<br>• SFR used by the real-time clock (RTC) and 12-bit interval timer cannot be written.<br>• The real-time clock (RTC) and 12-bit interval timer are in the reset status. |
| 1 | Enables clock supply.<br>• SFR used by the real-time clock (RTC) and 12-bit interval timer can be read and written. |

**Note** 32, 48, and 64-pin products only.

<R> **Cautions 1. When using the 12-bit interval timer, be sure to first set the RTCEN bit to 1 and then set the interval timer control register (ITMC), while oscillation of the count clock is stable. If RTCEN = 0, writing to the registers controlling the 12-bit interval timer is ignored, and, even if the register is read, only the default value is read (except the subsystem clock supply mode control register (OSMC)).**

**2. Clock supply to peripheral functions other than the real-time clock and 12-bit interval timer can be stopped in STOP mode or HALT mode when the subsystem clock is used, by setting the RTCLPC bit of the subsystem clock supply mode control register (OSMC) to 1.**

**3. Be sure to clear the following bits to 0.**

**25-pin products: bits 1, 3, 6**

**32, 48, 64-pin products: bits 1, 6**

### 8.3.2 Subsystem clock supply mode control register (OSMC)

The WUTMMCK0 bit can be used to select the 12-bit interval timer and real-time clock operation clock.

In addition, by stopping clock functions that are unnecessary, the RTCLPC bit can be used to reduce power consumption. For details about setting the RTCLPC bit, see **CHAPTER 5 CLOCK GENERATOR**.

The OSMC register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 8-3. Format of Subsystem Clock Supply Mode Control Register (OSMC)**

Address: F00F3H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| OSMC | RTCLPC | 0 | 0 | WUTMMCK0 | 0 | 0 | 0 | 0 |

| WUTMMCK0 | Selection of operation clock for real-time clock and 12-bit interval timer. |
|---|---|
| 0 | Subsystem clock ($f_{SUB}$) |
| 1 | Low-speed on-chip oscillator clock ($f_{IL}$) |

### 8.3.3 Interval timer control register (ITMC)

This register is used to set up the starting and stopping of the 12-bit interval timer operation and to specify the timer compare value.

The ITMC register can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0FFFH.

**Figure 8-4. Format of 12-bit Interval Timer Control Register (ITMC)**

Address: FFF90H After reset: 0FFFH R/W

| Symbol | 15 | 14 | 13 | 12 | 11 to 0 |
|---|---|---|---|---|---|
| ITMC | RINTE | 0 | 0 | 0 | ITCMP11 to ITCMP0 |

| RINTE | 12-bit interval timer operation control |
|---|---|
| 0 | Count operation stopped (count clear) |
| 1 | Count operation started |

| ITCMP11 to ITCMP0 | Specification of the 12-bit interval timer compare value |
|---|---|
| 001H | These bits generate an interrupt at the fixed cycle (count clock cycles x (ITCMP setting + 1)). |
| • • • | |
| FFFH | |
| 000H | Setting prohibit |
| Example interrupt cycles when 001H or FFFH is specified for ITCMP11 to ITCMP0<br>• ITCMP11 to ITCMP0 = 001H, count clock: when $f_{SUB}$ = 32.768 kHz<br>1/32.768 [kHz] × (1 + 1) = 0.06103515625 [ms] ≅ 61.03 [$\mu$s]<br>• ITCMP11 to ITCMP0 = FFFH, count clock: when $f_{SUB}$ = 32.768 kHz<br>1/32.768 [kHz] × (4095 + 1) = 125 [ms] | |

**Cautions 1. Before changing the RINTE bit from 1 to 0, use the interrupt mask flag register to disable the INTIT interrupt servicing. When the operation starts (from 0 to 1) again, clear the ITIF flag, and then enable the interrupt servicing.**

**2. The value read from the RINTE bit is applied one count clock cycle after setting the RINTE bit.**

**3. When setting the RINTE bit to start operation of the counter after returning from a standby mode and then shifting to a standby mode again, either confirm that the value written to the RINTE bit has been applied, or make sure that at least one count clock cycle elapses between returning from a standby mode and shifting to a standby mode again.**

**4. Only change the setting of the ITCMP11 to ITCMP0 bits when RINTE = 0.**
**However, it is possible to change the settings of the ITCMP11 to ITCMP0 bits at the same time as when changing RINTE from 0 to 1 or 1 to 0.**

## 8.4 12-bit Interval Timer Operation

### 8.4.1 12-bit interval timer operation timing

The count value specified for the ITCMP11 to ITCMP0 bits is used as an interval to operate as a 12-bit interval timer that repeatedly generates interrupt requests (INTIT).

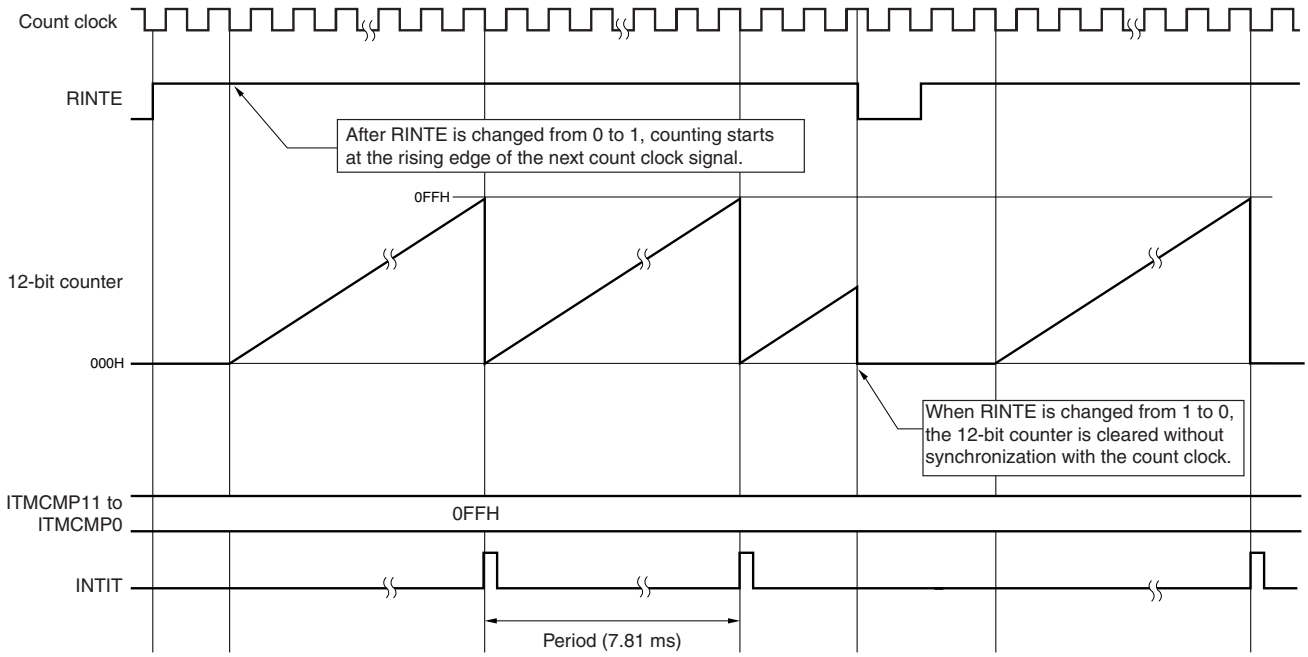
When the RINTE bit is set to 1, the 12-bit counter starts counting.

When the 12-bit counter value matches the value specified for the ITCMP11 to ITCMP0 bits, the 12-bit counter value is cleared to 0, counting continues, and an interrupt request signal (INTIT) is generated at the same time.

The basic operation of the 12-bit interval timer is as follows.

**Figure 8-5. 12-bit Interval Timer Operation Timing (ITCMP11 to ITCMP0 = 0FFH, count clock: $f_{SUB}$ = 32.768 kHz)**

<R>

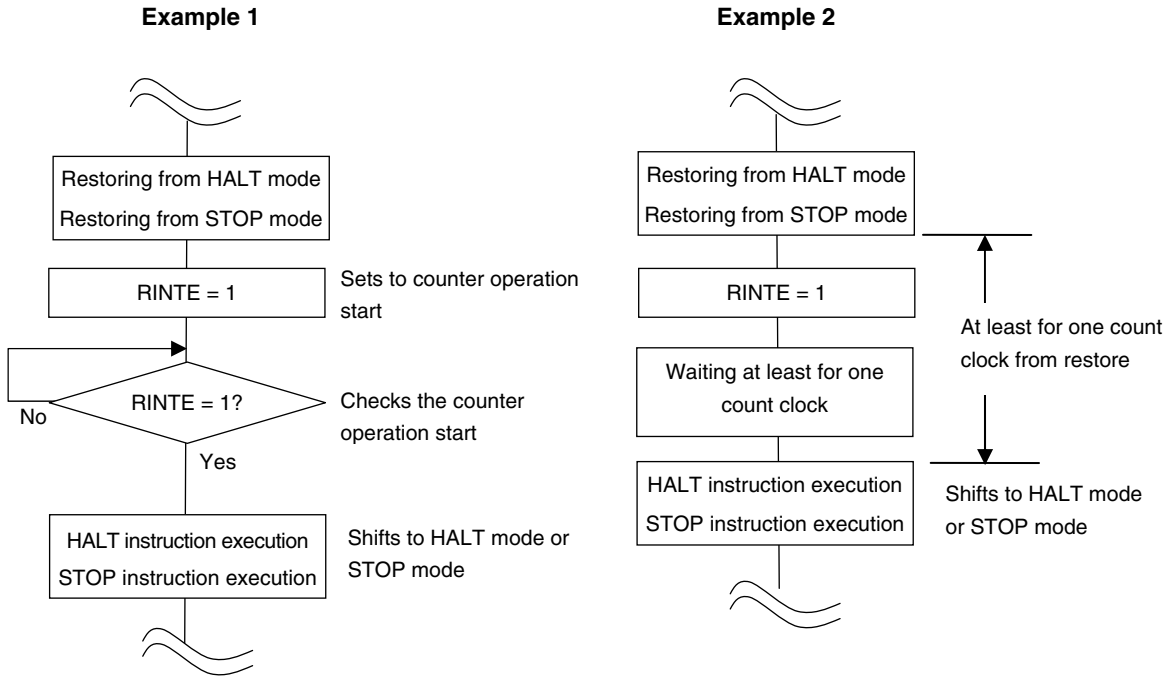


### 8.4.2 Starting counter operation after returning from HALT or STOP mode and then shifting to HALT or STOP mode again

When setting the RINTE bit to 1 to start operation of the counter after returning from the HALT or STOP mode and then shifting to the HALT or STOP mode again, either confirm that the value written to the RINTE bit (1) has been applied, or make sure that at least one count clock cycle elapses between returning from HALT or STOP mode and shifting to HALT or STOP mode again.

- After setting RINTE to 1, confirm that the RINTE bit value is actually 1 by polling the bit, and then shift to the HALT or STOP mode (see **Example 1** in **Figure 8-6**).
- After setting RINTE to 1, wait for at least one cycle of the count clock to elapse before shifting to the HALT or STOP mode (see **Example 2** in **Figure 8-6**).

**Figure 8-6. Shifting to HALT or STOP Mode after Setting RINTE to 1**



# CHAPTER 9 CLOCK OUTPUT/BUZZER OUTPUT CONTROLLER

The number of output pins of the clock output and buzzer output controllers differs, depending on the product.

| Output pin | 25-pin | 32, 48, 64-pin |
|---|---|---|
| PCLBUZ0 | √ | √ |
| PCLBUZ1 | – | √ |

**Caution Most of the following descriptions in this chapter use the 64-pin as an example.**

## 9.1 Functions of Clock Output/Buzzer Output Controller

<R> The clock output controller is intended for clock output for supply to peripheral ICs. Buzzer output is a function to output a square wave of buzzer frequency.

One pin can be used to output a clock or buzzer sound.

Two output pins, PCLBUZ0 and PCLBUZ1, are available.

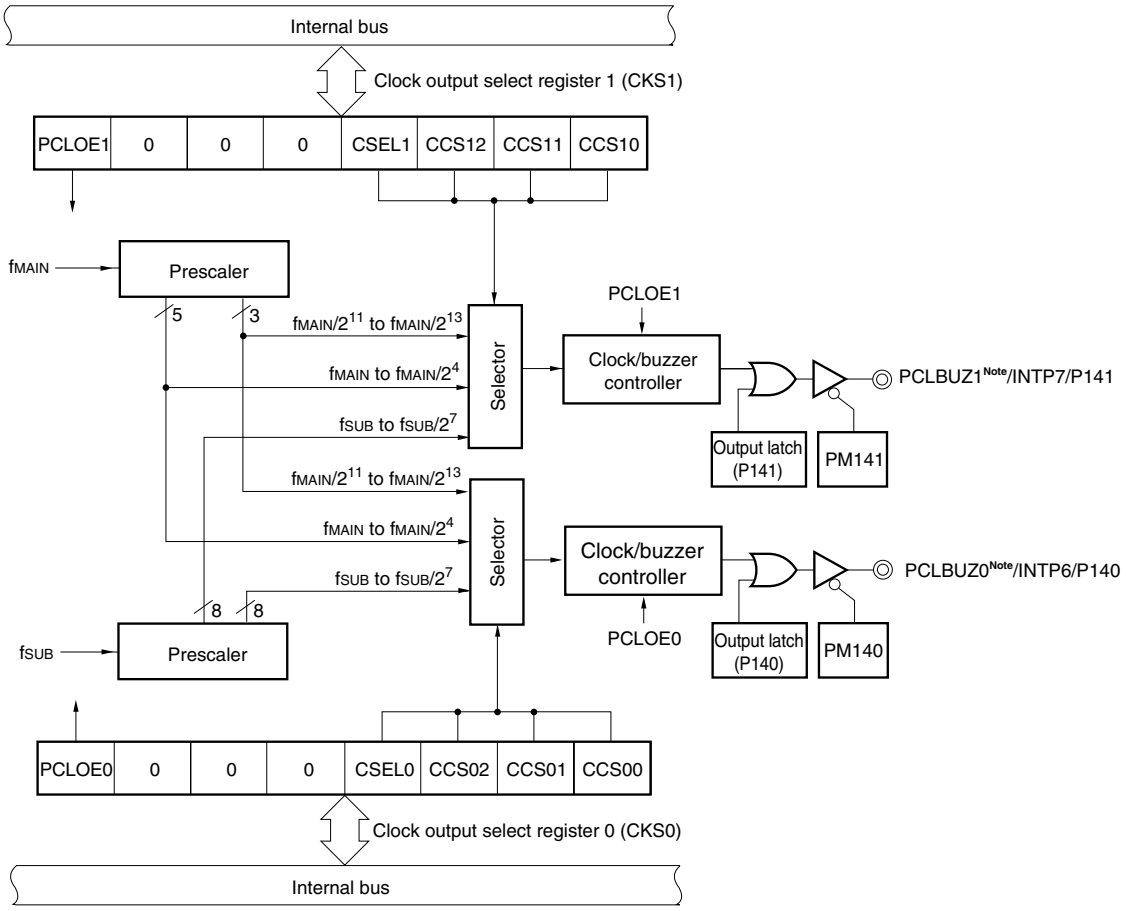
The PCLBUZn pin outputs a clock selected by clock output select register n (CKSn).

Figure 9-1 shows the block diagram of clock output/buzzer output controller.

<R> **Caution It is not possible to output the subsystem clock ($f_{SUB}$) from the PCLBUZn pin while the RTCLPC bit of the subsystem clock supply mode control register (OSMC), which controls the supply of the subsystem clock, is set to 1 and moreover while HALT mode is set with the subsystem clock ($f_{SUB}$) selected as CPU clock.**

**Remark** n = 0, 1

**Figure 9-1. Block Diagram of Clock Output/Buzzer Output Controller**




**Note** For output frequencies available from PCLBUZ0 and PCLBUZ1, see **29.4** or **30.4 AC Characteristics**.

**Remark** The clock output/buzzer output pins in above diagram shows the information of 64-pins products.

## 9.2 Configuration of Clock Output/Buzzer Output Controller

The clock output/buzzer output controller includes the following hardware.

**Table 9-1. Configuration of Clock Output/Buzzer Output Controller**

| Item | Configuration |
|---|---|
| Control registers | Clock output select registers n (CKSn)<br>Port mode register 14 (PM14)<br>Port register 14 (P14) |

## 9.3 Registers Controlling Clock Output/Buzzer Output Controller

The following two registers are used to control the clock output/buzzer output controller.

- Clock output select registers n (CKSn)
- Port mode register 14 (PM14)
- Port register 14 (P14)

### 9.3.1 Clock output select registers n (CKSn)

These registers set output enable/disable for clock output or for the buzzer frequency output pin (PCLBUZn), and set the output clock.

Select the clock to be output from the PCLBUZn pin by using the CKSn register.

The CKSn register are set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H.

**Figure 9-2. Format of Clock Output Select Register n (CKSn)**

Address: FFFA5H (CKS0), FFFA6H (CKS1) After reset: 00H R/W

| Symbol | <7> | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CKSn | PCLOEn | 0 | 0 | 0 | CSELn | CCSn2 | CCSn1 | CCSn0 |

| PCLOEn | PCLBUZn pin output enable/disable specification |
|---|---|
| 0 | Output disable (default) |
| 1 | Output enable |

| CSELn | CCSn2 | CCSn1 | CCSn0 | PCLBUZn pin output clock selection | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | $f_{MAIN}$ = 5 MHz | $f_{MAIN}$ = 10 MHz | $f_{MAIN}$ = 20 MHz | $f_{MAIN}$ = 32 MHz |
| 0 | 0 | 0 | 0 | $f_{MAIN}$ | 5 MHz | Setting prohibited[Note] | Setting prohibited[Note] | Setting prohibited[Note] |
| 0 | 0 | 0 | 1 | $f_{MAIN}/2$ | 2.5 MHz | 5 MHz | Setting prohibited[Note] | Setting prohibited[Note] |
| 0 | 0 | 1 | 0 | $f_{MAIN}/2^2$ | 1.25 MHz | 2.5 MHz | 5 MHz | 8 MHz[Note] |
| 0 | 0 | 1 | 1 | $f_{MAIN}/2^3$ | 625 kHz | 1.25 MHz | 2.5 MHz | 4 MHz |
| 0 | 1 | 0 | 0 | $f_{MAIN}/2^4$ | 312.5 kHz | 625 kHz | 1.25 MHz | 2 MHz |
| 0 | 1 | 0 | 1 | $f_{MAIN}/2^{11}$ | 2.44 kHz | 4.88 kHz | 9.76 kHz | 15.63 kHz |
| 0 | 1 | 1 | 0 | $f_{MAIN}/2^{12}$ | 1.22 kHz | 2.44 kHz | 4.88 kHz | 7.81 kHz |
| 0 | 1 | 1 | 1 | $f_{MAIN}/2^{13}$ | 610 Hz | 1.22 kHz | 2.44 kHz | 3.91 kHz |
| 1 | 0 | 0 | 0 | $f_{SUB}$ | 32.768 kHz | | | |
| 1 | 0 | 0 | 1 | $f_{SUB}/2$ | 16.384 kHz | | | |
| 1 | 0 | 1 | 0 | $f_{SUB}/2^2$ | 8.192 kHz | | | |
| 1 | 0 | 1 | 1 | $f_{SUB}/2^3$ | 4.096 kHz | | | |
| 1 | 1 | 0 | 0 | $f_{SUB}/2^4$ | 2.048 kHz | | | |
| 1 | 1 | 0 | 1 | $f_{SUB}/2^5$ | 1.024 kHz | | | |
| 1 | 1 | 1 | 0 | $f_{SUB}/2^6$ | 512 Hz | | | |
| 1 | 1 | 1 | 1 | $f_{SUB}/2^7$ | 256 Hz | | | |

**Note** Use the output clock within a range of 8 MHz. Furthermore, when using the output clock at 2.7 V ≤ $V_{DD}$ < 4.0 V, can be use it within 8 MHz only. See **29.4** or **30.4 AC Characteristics** for details.

**Cautions 1. Change the output clock after disabling clock output (PCLOEn = 0).**

<R> **2. To shift to STOP mode when the main system clock is selected (CSELn = 0), set PCLOEn = 0 before executing the STOP instruction. When the subsystem clock is selected (CSELn = 1), PCLOEn = 1 can be set because the clock can be output while the RTCLPC bit of the subsystem clock supply mode control register (OSMC) is set to 0 and moreover while STOP mode is set.**

<R> **3. It is not possible to output the subsystem clock ($f_{SUB}$) from the PCLBUZn pin while the RTCLPC bit of the subsystem clock supply mode control register (OSMC), which controls the supply of the subsystem clock, is set to 1 and moreover while HALT mode is set with the subsystem clock ($f_{SUB}$) selected as CPU clock.**

**Remarks 1.** n = 0, 1

**2.** $f_{MAIN}$: Main system clock frequency

$f_{SUB}$: Subsystem clock frequency

<R> **9.3.2 Registers controlling port functions of pins to be used for clock or buzzer output**

Using a port pin for clock or buzzer output requires setting of the registers that control the port functions multiplexed on the target pin (port mode register (PMxx), port register (Pxx)). For details, see **4.3.1 Port mode registers (PMxx)** and **4.3.2 Port registers (Pxx)**.

Specifically, using a port pin with a multiplexed clock or buzzer output function (e.g. P140/INTP6/PCLBUZ0, P141/INTP7/PCLBUZ1) for clock or buzzer output, requires setting the corresponding bits in the port mode register (PMxx) and port register (Pxx) to 0.

Example: When P140/INTP6/PCLBUZ0 is to be used for clock or buzzer output
Set the PM140 bit of port mode register 14 to 0.
Set the P140 bit of port register 14 to 0.

## 9.4 Operations of Clock Output/Buzzer Output Controller

One pin can be used to output a clock or buzzer sound.

The PCLBUZ0 pin outputs a clock/buzzer selected by the clock output select register 0 (CKS0).

The PCLBUZ1 pin outputs a clock/buzzer selected by the clock output select register 1 (CKS1).

### 9.4.1 Operation as output pin

The PCLBUZn pin is output as the following procedure.

<R> <1> Set 0 in the bit of the port mode register (PMxx) and port register (Pxx) which correspond to the port which has a pin used as the PCLBUZn pin.

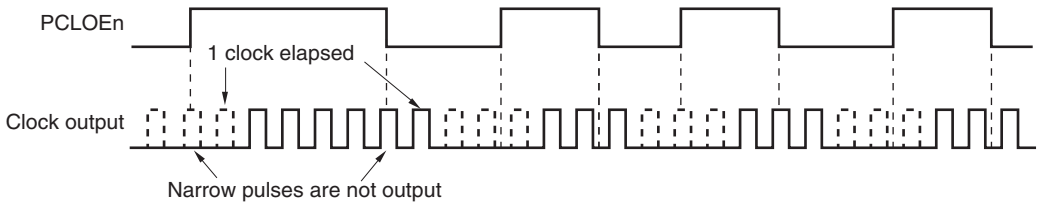
<2> Select the output frequency with bits 0 to 3 (CCSn0 to CCSn2, CSELn) of the clock output select register (CKSn) of the PCLBUZn pin (output in disabled status).

<3> Set bit 7 (PCLOEn) of the CKSn register to 1 to enable clock/buzzer output.

**Remarks 1.** The controller used for outputting the clock starts or stops outputting the clock one clock after enabling or disabling clock output (PCLOEn bit) is switched. At this time, pulses with a narrow width are not output. Figure 9-3 shows enabling or stopping output using the PCLOEn bit and the timing of outputting the clock.

**2.** n = 0, 1

<R> **Figure 9-3. Timing of Outputting Clock from PCLBUZn Pin**




## 9.5 Cautions of Clock Output/Buzzer Output Controller

<R> When the main system clock is selected for the PCLBUZn output (CSEL = 0), if STOP or HALT mode is entered within 1.5 clock cycles output from the PCLBUZn pin after the output is disabled (PCLOEn = 0), the PCLBUZn output width becomes shorter.

# CHAPTER 10 WATCHDOG TIMER

## 10.1 Functions of Watchdog Timer

<R> The counting operation of the watchdog timer is set by the option byte (000C0H).

The watchdog timer operates on the low-speed on-chip oscillator clock ($f_{IL}$).

The watchdog timer is used to detect an inadvertent program loop. If a program loop is detected, an internal reset signal is generated.

Program loop is detected in the following cases.

- If the watchdog timer counter overflows
- If a 1-bit manipulation instruction is executed on the watchdog timer enable register (WDTE)
- If data other than “ACH” is written to the WDTE register
- If data is written to the WDTE register during a window close period

When a reset occurs due to the watchdog timer, bit 4 (WDTRF) of the reset control flag register (RESF) is set to 1. For details of the RESF register, see **CHAPTER 19 RESET FUNCTION**.

When 75% + 1/2/$f_{IL}$ of the overflow time is reached, an interval interrupt can be generated.

## 10.2 Configuration of Watchdog Timer

The watchdog timer includes the following hardware.

<R>

**Table 10-1. Configuration of Watchdog Timer**

| Item | Configuration |
|---|---|
| Counter | Internal counter (17 bits) |
| Control register | Watchdog timer enable register (WDTE) |

How the counter operation is controlled, overflow time, window open period, and interval interrupt are set by the option byte.

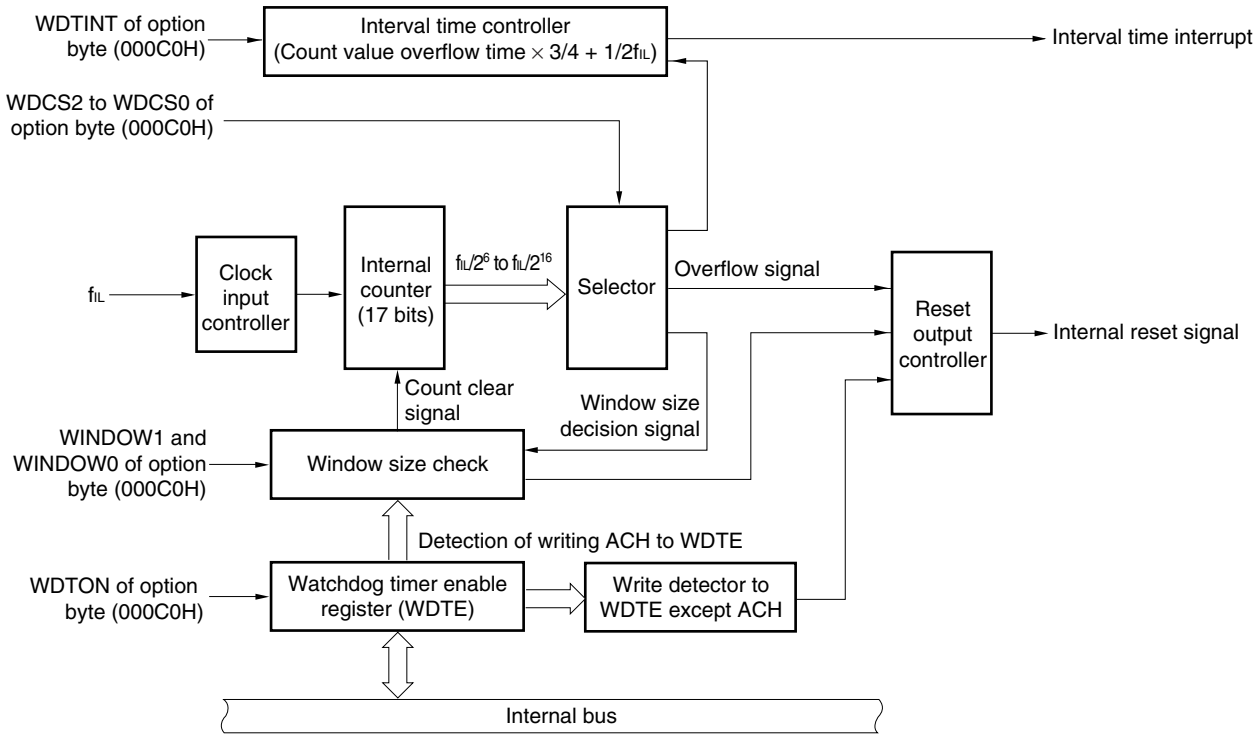
**Table 10-2. Setting of Option Bytes and Watchdog Timer**

| Setting of Watchdog Timer | Option Byte (000C0H) |
|---|---|
| Watchdog timer interval interrupt | Bit 7 (WDTINT) |
| Window open period | Bits 6 and 5 (WINDOW1, WINDOW0) |
| Controlling counter operation of watchdog timer | Bit 4 (WDTON) |
| Overflow time of watchdog timer | Bits 3 to 1 (WDCS2 to WDCS0) |
| Controlling counter operation of watchdog timer (in HALT/STOP mode) | Bit 0 (WDSTBYON) |

**Remark** For the option byte, see **CHAPTER 24 OPTION BYTE**.

<R>

**Figure 10-1. Block Diagram of Watchdog Timer**




**Remark** $f_{IL}$: Low-speed on-chip oscillator clock

## 10.3 Register Controlling Watchdog Timer

The watchdog timer is controlled by the watchdog timer enable register (WDTE).

### 10.3.1 Watchdog timer enable register (WDTE)

Writing “ACH” to the WDTE register clears the watchdog timer counter and starts counting again.

This register can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets this register to 9AH or 1AH[Note].

**Figure 10-2. Format of Watchdog Timer Enable Register (WDTE)**

Address: FFFABH After reset: 9AH/1AH[Note] R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| WDTE | | | | | | | | |

**Note** The WDTE register reset value differs depending on the WDTON bit setting value of the option byte (000C0H). To operate watchdog timer, set the WDTON bit to 1.

| WDTON Bit Setting Value | WDTE Register Reset Value |
|---|---|
| 0 (watchdog timer count operation disabled) | 1AH |
| 1 (watchdog timer count operation enabled) | 9AH |

**Cautions 1. If a value other than “ACH” is written to the WDTE register, an internal reset signal is generated.**

**2. If a 1-bit memory manipulation instruction is executed for the WDTE register, an internal reset signal is generated.**

**3. The value read from the WDTE register is 9AH/1AH (this differs from the written value (ACH)).**

## 10.4 Operation of Watchdog Timer

### 10.4.1 Controlling operation of watchdog timer

1. When the watchdog timer is used, its operation is specified by the option byte (000C0H).

   - Enable counting operation of the watchdog timer by setting bit 4 (WDTON) of the option byte (000C0H) to 1 (the counter starts operating after a reset release) (for details, see **CHAPTER 24**).

| WDTON | Watchdog Timer Counter |
|---|---|
| 0 | Counter operation disabled (counting stopped after reset) |
| 1 | Counter operation enabled (counting started after reset) |

   - Set an overflow time by using bits 3 to 1 (WDCS2 to WDCS0) of the option byte (000C0H) (for details, see **10.4.2** and **CHAPTER 24**).
   - Set a window open period by using bits 6 and 5 (WINDOW1 and WINDOW0) of the option byte (000C0H) (for details, see **10.4.3** and **CHAPTER 24**).

2. After a reset release, the watchdog timer starts counting.
3. By writing “ACH” to the watchdog timer enable register (WDTE) after the watchdog timer starts counting and before the overflow time set by the option byte, the watchdog timer is cleared and starts counting again.
4. After that, write the WDTE register the second time or later after a reset release during the window open period. If the WDTE register is written during a window close period, an internal reset signal is generated.
5. If the overflow time expires without “ACH” written to the WDTE register, an internal reset signal is generated.
   An internal reset signal is generated in the following cases.

   - If a 1-bit manipulation instruction is executed on the WDTE register
   - If data other than “ACH” is written to the WDTE register

**Cautions 1. When data is written to the watchdog timer enable register (WDTE) for the first time after reset release, the watchdog timer is cleared in any timing regardless of the window open time, as long as the register is written before the overflow time, and the watchdog timer starts counting again.**

<R> **2. After "ACH" is written to the WDTE register, an error of up to 2 clocks ($f_{IL}$) may occur before the watchdog timer is cleared.**

**3. The watchdog timer can be cleared immediately before the count value overflows.**

**Cautions 4. The operation of the watchdog timer in the HALT, STOP, and SNOOZE modes differs as follows depending on the set value of bit 0 (WDSTBYON) of the option byte (000C0H).**

| | WDSTBYON = 0 | WDSTBYON = 1 |
|---|---|---|
| In HALT mode | Watchdog timer operation stops. | Watchdog timer operation continues. |
| In STOP mode | | |
| In SNOOZE mode | | |

**If WDSTBYON = 0, the watchdog timer resumes counting after the HALT or STOP mode is released. At this time, the counter is cleared to 0 and counting starts.**
**When operating with the X1 oscillation clock after releasing the STOP mode, the CPU starts operating after the oscillation stabilization time has elapsed.**
**Therefore, if the period between the STOP mode release and the watchdog timer overflow is short, an overflow occurs during the oscillation stabilization time, causing a reset.**
**Consequently, set the overflow time in consideration of the oscillation stabilization time when operating with the X1 oscillation clock and when the watchdog timer is to be cleared after the STOP mode release by an interval interrupt.**

### 10.4.2 Setting overflow time of watchdog timer

Set the overflow time of the watchdog timer by using bits 3 to 1 (WDCS2 to WDCS0) of the option byte (000C0H).

If an overflow occurs, an internal reset signal is generated. The present count is cleared and the watchdog timer starts counting again by writing “ACH” to the watchdog timer enable register (WDTE) during the window open period before the overflow time.

The following overflow times can be set.

**Table 10-3. Setting of Overflow Time of Watchdog Timer**

| WDCS2 | WDCS1 | WDCS0 | Overflow Time of Watchdog Timer ($f_{IL}$ = 17.25 kHz (MAX.)) |
|---|---|---|---|
| 0 | 0 | 0 | $2^6/f_{IL}$ (3.71 ms) |
| 0 | 0 | 1 | $2^7/f_{IL}$ (7.42 ms) |
| 0 | 1 | 0 | $2^8/f_{IL}$ (14.84 ms) |
| 0 | 1 | 1 | $2^9/f_{IL}$ (29.68 ms) |
| 1 | 0 | 0 | $2^{11}/f_{IL}$ (118.72 ms) |
| <R> 1 | 0 | 1 | $2^{13}/f_{IL}$ (474.89 ms) |
| <R> 1 | 1 | 0 | $2^{14}/f_{IL}$ (949.79 ms) |
| <R> 1 | 1 | 1 | $2^{16}/f_{IL}$ (3799.18 ms) |

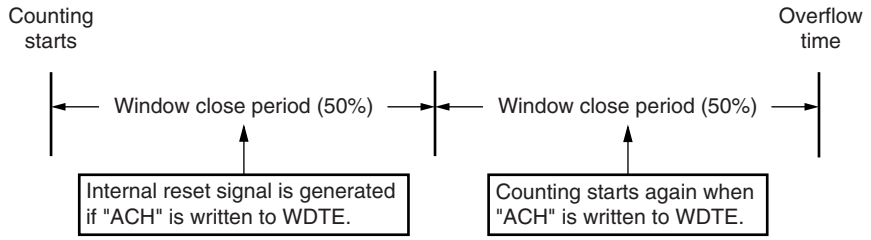
**Remark** $f_{IL}$: Low-speed on-chip oscillator clock frequency

### 10.4.3 Setting window open period of watchdog timer

Set the window open period of the watchdog timer by using bits 6 and 5 (WINDOW1, WINDOW0) of the option byte (000C0H). The outline of the window is as follows.

- If “ACH” is written to the watchdog timer enable register (WDTE) during the window open period, the watchdog timer is cleared and starts counting again.
- Even if “ACH” is written to the WDTE register during the window close period, an abnormality is detected and an internal reset signal is generated.

**Example**: If the window open period is 50%




**Caution When data is written to the WDTE register for the first time after reset release, the watchdog timer is cleared in any timing regardless of the window open time, as long as the register is written before the overflow time, and the watchdog timer starts counting again.**

The window open period can be set is as follows.

**Table 10-4. Setting Window Open Period of Watchdog Timer**

| WINDOW1 | WINDOW0 | Window Open Period of Watchdog Timer |
|---|---|---|
| 0 | 0 | Setting prohibited |
| 0 | 1 | 50% |
| 1 | 0 | 75% |
| 1 | 1 | 100% |

**Caution When bit 0 (WDSTBYON) of the option byte (000C0H) = 0, the window open period is 100% regardless of the values of the WINDOW1 and WINDOW0 bits.**

**Remark** If the overflow time is set to $2^9/f_{IL}$, the window close time and open time are as follows.

| | Setting of Window Open Period | | |
|---|---|---|---|
| | 50% | 75% | 100% |
| Window close time | 0 to 20.08 ms | 0 to 10.04 ms | None |
| Window open time | 20.08 to 29.68 ms | 10.04 to 29.68 ms | 0 to 29.68 ms |

<When window open period is 50%>

- Overflow time:
  $2^9/f_{IL}$ (MAX.) = $2^9$/17.25 kHz = 29.68 ms
- Window close time:
  0 to $2^9/f_{IL}$ (MIN.) × (1 − 0.5) = 0 to $2^9$/12.75 kHz × 0.5 = 0 to 20.08 ms
- Window open time:
  $2^9/f_{IL}$ (MIN.) × (1 − 0.5) to $2^9/f_{IL}$ (MAX.) = $2^9$/12.75 kHz × 0.5 to $2^9$/17.25 kHz = 20.08 to 29.68 ms

### 10.4.4 Setting watchdog timer interval interrupt

Depending on the setting of bit 7 (WDTINT) of an option byte (000C0H), an interval interrupt (INTWDTI) can be generated when 75% + 1/2$f_{IL}$ of the overflow time is reached.

**Table 10-5. Setting of Watchdog Timer Interval Interrupt**

| WDTINT | Use of Watchdog Timer Interval Interrupt |
|---|---|
| 0 | Interval interrupt is used. |
| 1 | Interval interrupt is generated when 75% + 1/2$f_{IL}$ of overflow time is reached. |

**Caution When operating with the X1 oscillation clock after releasing the STOP mode, the CPU starts operating after the oscillation stabilization time has elapsed.**
**Therefore, if the period between the STOP mode release and the watchdog timer overflow is short, an overflow occurs during the oscillation stabilization time, causing a reset.**
**Consequently, set the overflow time in consideration of the oscillation stabilization time when operating with the X1 oscillation clock and when the watchdog timer is to be cleared after the STOP mode release by an interval interrupt.**

**Remark** The watchdog timer continues counting even after INTWDTI is generated (until ACH is written to the watchdog timer enable register (WDTE)). If ACH is not written to the WDTE register before the overflow time, an internal reset signal is generated.

# CHAPTER 11 A/D CONVERTER

The number of analog input channels of the A/D converter differs, depending on the product.

| | | | 25-pin | 32-pin | 48-pin | 64-pin |
|---|---|---|---|---|---|---|
| Analog input channels | | Total | 13 ch | 18 ch | 24 ch | 28 ch |
| | High accuracy channel | Pins based on input buffer power supply $AV_{DD}$ | 4 ch (ANI0 to ANI3) | 5 ch (ANI0 to ANI4) | 9 ch (ANI0 to ANI8) | 13 ch (ANI0 to ANI12) |
| | Standard channel | Pins based on input buffer power supply $V_{DD}$ | 9 ch (ANI16 to ANI18, ANI20, ANI21, ANI25 to ANI27, ANI29) | 13 ch (ANI16 to ANI24, ANI26 to ANI29) | 15 ch (ANI16 to ANI30) | – |
| | | Pins based on input buffer power supply $EV_{DD0}$ | – | – | – | 15 ch (ANI16 to ANI30) |

**Remark** Most of the following descriptions in this chapter use the 64-pin products as an example. With 25 to 48-pin products, replace $EV_{DD0}$ with $V_{DD}$, or replace $EV_{SS0}$ with $V_{SS}$.

## 11.1 Function of A/D Converter

The A/D converter converts analog input signals into digital values, and is configured to control analog inputs, including up to 28 channels of A/D converter analog inputs (ANI0 to ANI12 and ANI16 to ANI30). 12-bit resolution mode or 8-bit resolution mode can also be selected by using the ADTYP bit of A/D converter mode register 2 (ADM2).

The A/D converter has the following function.

- **12-bit/8-bit resolution A/D conversion**
  A/D conversion is carried out repeatedly for one analog input channel selected from ANI0 to ANI12 and ANI16 to ANI30. Each time an A/D conversion operation ends, an interrupt request signal (INTAD) is generated (when in the select mode).

<R> **Caution The valid resolution differs depending on the voltage conditions of $AV_{DD}$ and $AV_{REFP}$. For details, see 29.6.1 A/D converter characteristics.**

<R> **Remark** When using the converter with a resolution of 10 bits, select the 12-bit resolution mode (ADTYP = 0). Use the higher 10 bits of the conversion result. Do not use the lower 2 bits.

Various A/D conversion modes can be specified by using the mode combinations below.

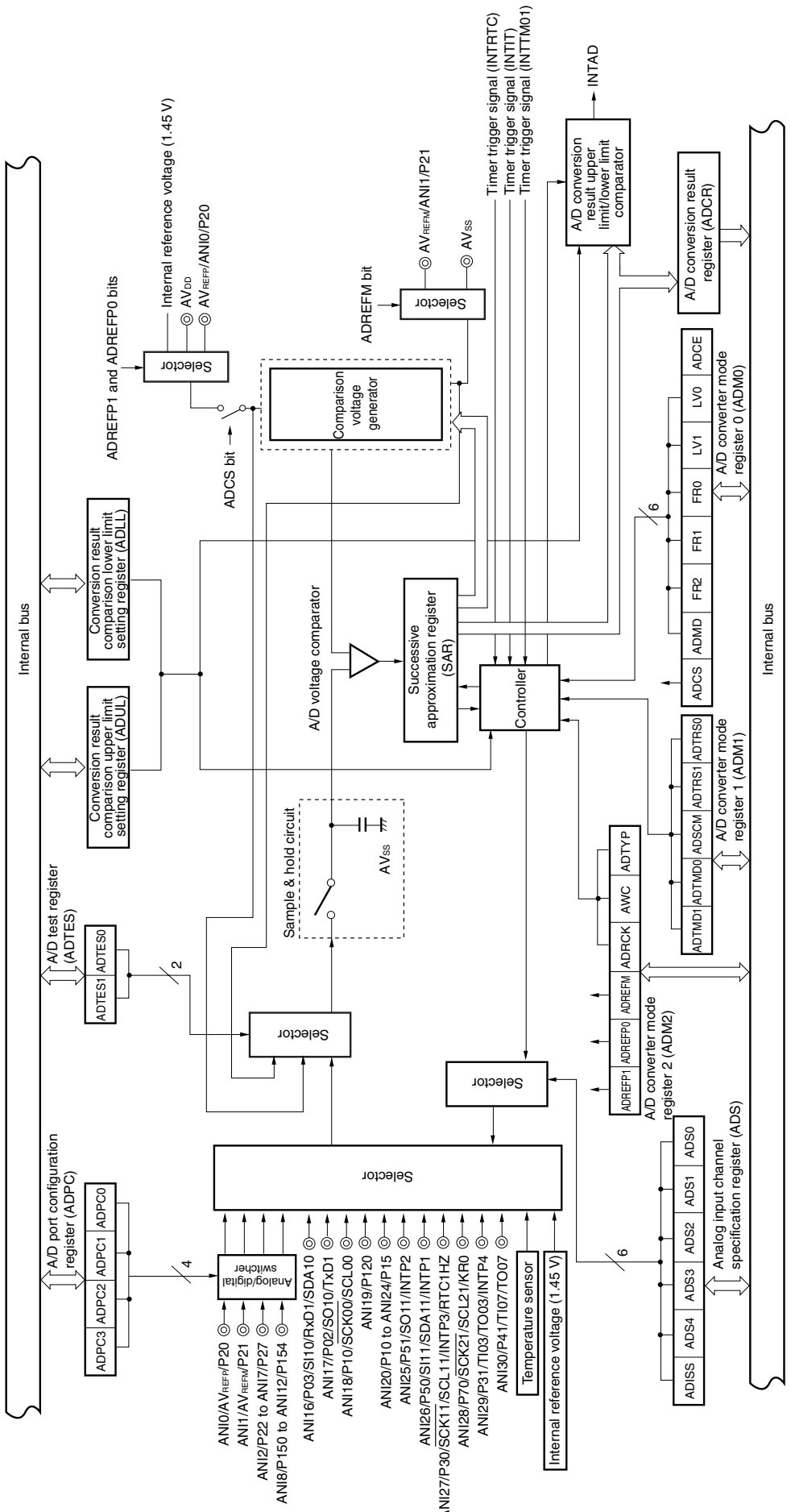
| | | |
|---|---|---|
| Trigger mode | Software trigger | Conversion is started by software manipulation. |
| | Hardware trigger no-wait mode | Conversion is started by detecting a hardware trigger. |
| | Hardware trigger wait mode | The power is turned on by detecting a hardware trigger while the system is off and in the conversion standby state, and conversion is then started automatically after the A/D power supply stabilization wait time passes.<br>When using the SNOOZE mode function, specify the hardware trigger wait mode. |
| Channel selection mode | Select mode | A/D conversion is performed on the analog input of one selected channel. |
| | Scan mode | A/D conversion is performed on the analog input of four channels in order. Four consecutive channels can be selected from ANI0 to ANI12 as analog input channels. |
| Conversion operation mode | One-shot conversion mode | A/D conversion is performed on the selected channel once. |
| | Sequential conversion mode | A/D conversion is sequentially performed on the selected channels until it is stopped by software. |

<R>

| Operation Mode[Note] | Number of Sampling Clock | |
|---|---|---|
| Normal 1 | 11 $f_{AD}$ | Set a value to the number of sampling clocks, at which the sampling capacitor is fully charged, depending on the output impedance of the analog input source. |
| Normal 2 | 23 $f_{AD}$ | |
| Low-voltage 1 | 33 $f_{AD}$ | |
| Low-voltage 2 | 187 $f_{AD}$ | |

**Note** The operation modes selectable differ depending on the analog input channel, $AV_{DD}$ voltage, trigger mode, and $f_{CLK}$. For details, **Table 11-3 A/D Conversion Time Selection**.

**Figure 11-1. Block Diagram of A/D Converter**




**Remark** Analog input pins in figure 11-1 when a 64-pin product is used.

## 11.2 Configuration of A/D Converter

The A/D converter includes the following hardware.

**(1) ANI0 to ANI12 and ANI16 to ANI30 pins**

These are the analog input pins of the 28 channels of the A/D converter. They input analog signals to be converted into digital signals. Pins other than the one selected as the analog input pin can be used as I/O port pins.

**(2) Sample & hold circuit**

The sample & hold circuit samples each of the analog input voltages sequentially sent from the input circuit, and sends them to the A/D voltage comparator. This circuit also holds the sampled analog input voltage during A/D conversion.

**(3) A/D voltage comparator**

This A/D voltage comparator compares output from the voltage tap of the comparison voltage generator with the sampled voltage value.

If the analog input voltage is found to be greater than the reference voltage (1/2 $AV_{REF}$) as a result of the comparison, the most significant bit (MSB) of the successive approximation register (SAR) is set. If the analog input voltage is less than the reference voltage (1/2 $AV_{REF}$), the MSB bit of the SAR is reset.

After that, bit 10 of the SAR register is automatically set, and the next comparison is made. The voltage tap of the comparison voltage generator is selected by the value of bit 11, to which the result has been already set.

Bit 11 = 0: (1/4 $AV_{REF}$)
Bit 11 = 1: (3/4 $AV_{REF}$)

The voltage tap of the comparison voltage generator and the analog input voltage are compared and bit 10 of the SAR register is manipulated according to the result of the comparison.

Analog input voltage ≥ Voltage tap of comparison voltage generator: Bit 10 = 1
Analog input voltage ≤ Voltage tap of comparison voltage generator: Bit 10 = 0

Comparison is continued like this to bit 0 of the SAR register.

When performing A/D conversion at a resolution of 8 bits, the comparison continues until bit 4 of the SAR register.

**Remark** $AV_{REF}$: The + side reference voltage of the A/D converter.
(This can be selected from $AV_{REFP}$, the internal reference voltage (1.45 V), and $AV_{DD}$.)

**(4) Comparison voltage generator**

The comparison voltage generator generates the comparison voltage input from an analog input pin.

**(5) Successive approximation register (SAR)**

The SAR register is a register that sets voltage tap data whose values from the comparison voltage generator match the voltage values of the analog input pins, 1 bit at a time starting from the most significant bit (MSB).

If data is set in the SAR register all the way to the least significant bit (LSB) (end of A/D conversion), the contents of the SAR register (conversion results) are held in the A/D conversion result register (ADCR). When all the specified A/D conversion operations have ended, an A/D conversion end interrupt request signal (INTAD) is generated.

**(6) 12-bit A/D conversion result register (ADCR)**

The A/D conversion result is loaded from the successive approximation register to this register each time A/D conversion is completed, and the ADCR register holds the A/D conversion result in its lower 12 bits (the higher 4 bits are fixed to 0).

**(7) 8-bit A/D conversion result register (ADCRH)**

The A/D conversion result is loaded from the successive approximation register to this register each time A/D conversion is completed, and the ADCRH register stores the higher 8 bits of the A/D conversion result.

**(8) Controller**

This circuit controls the conversion time of an input analog signal that is to be converted into a digital signal, as well as starting and stopping of the conversion operation. When A/D conversion has been completed, this controller generates an interrupt request signal (INTAD) through the A/D conversion result upper limit/lower limit comparator.

**(9) $AV_{REFP}$ pin**

This pin inputs an external reference voltage ($AV_{REFP}$).

If using $AV_{REFP}$ as the + side reference voltage of the A/D converter, set the ADREFP1 and ADREFP0 bits of A/D converter mode register 2 (ADM2) to 0 and 1, respectively.

The analog signals input to ANI2 to ANI12 and ANI16 to ANI30 are converted to digital signals based on the voltage applied between $AV_{REFP}$ and the − side reference voltage ($AV_{REFM}$/$AV_{SS}$).

In addition to $AV_{REFP}$, it is possible to select $AV_{DD}$, or the internal reference voltage (1.45 V) as the + side reference voltage of the A/D converter.

**(10) $AV_{REFM}$ pin**

This pin inputs an external reference voltage ($AV_{REFM}$). If using $AV_{REFM}$ as the − side reference voltage of the A/D converter, set the ADREFM bit of the ADM2 register to 1.

In addition to $AV_{REFM}$, it is possible to select $AV_{SS}$ as the − side reference voltage of the A/D converter.

## 11.3 Registers Used in A/D Converter

The A/D converter is controlled by the following registers.

- Peripheral enable register 0 (PER0)
- A/D converter mode register 0 (ADM0)
- A/D converter mode register 1 (ADM1)
- A/D converter mode register 2 (ADM2)
- 12-bit A/D conversion result register (ADCR)
- 8-bit A/D conversion result register (ADCRH)
- Analog input channel specification register (ADS)
- Conversion result comparison upper limit setting register (ADUL)
- Conversion result comparison lower limit setting register (ADLL)
- A/D test register (ADTES)
- A/D port configuration register (ADPC)
- Port mode control registers 0, 1, 3 to 5, 7, and 12 (PMC0, PMC1, PMC3 to PMC5, PMC7, PMC12)
- Port mode registers 0 to 5, 7, 12, and 15 (PM0 to PM5, PM7, PM12, PM15)

### 11.3.1 Peripheral enable register 0 (PER0)

This register is used to enable or disable supplying the clock to the peripheral hardware. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

When the A/D converter is used, be sure to set bit 5 (ADCEN) of this register to 1.

The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 11-2. Format of Peripheral Enable Register 0 (PER0)**

Address: F00F0H After reset: 00H R/W

| Symbol | <7> | 6 | <5> | <4> | <3> | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PER0 | RTCEN | 0 | ADCEN | IICA0EN | SAU1EN[Note] | SAU0EN | 0 | TAU0EN |

| ADCEN | Control of A/D converter input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFR used by the A/D converter cannot be written.<br>• The A/D converter is in the reset status. |
| 1 | Enables input clock supply.<br>• SFR used by the A/D converter can be read/written. |

**Note** 32-, 48-, 64-pin products only

<R> **Cautions 1. When setting the A/D converter, be sure to set the following registers first while the ADCEN bit is set to 1. If ADCEN = 0, the values of the A/D converter control registers are cleared to their initial values and writing to them is ignored (except for port mode registers 0 to 5, 7, 12, and 15 (PM0 to PM5, PM7, PM12, PM15), port mode control registers 0, 1, 3 to 5, 7, and 12 (PMC0, PMC1, PMC3 to PMC5, PMC7, PMC12), and A/D port configuration register (ADPC)).**

- **A/D converter mode register 0 (ADM0)**
- **A/D converter mode register 1 (ADM1)**
- **A/D converter mode register 2 (ADM2)**
- **12-bit A/D conversion result register (ADCR)**
- **8-bit A/D conversion result register (ADCRH)**
- **Analog input channel specification register (ADS)**
- **Conversion result comparison upper limit setting register (ADUL)**
- **Conversion result comparison lower limit setting register (ADLL)**
- **A/D test register (ADTES)**

**2. Be sure to clear the following bits to 0.**

**25-pin products: bits 1, 3, 6**

**32, 48, 64-pin products: bits 1, 6**

### 11.3.2 A/D converter mode register 0 (ADM0)

This register sets the conversion time for analog input to be A/D converted, and starts/stops conversion.

The ADM0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 11-3. Format of A/D Converter Mode Register 0 (ADM0)**

Address: FFF30H After reset: 00H R/W

| Symbol | <7> | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| ADM0 | ADCS | ADMD | FR2 Note 1 | FR1 Note 1 | FR0 Note 1 | LV1 Note 1 | LV0 Note 1 | ADCE |

| ADCS | A/D conversion operation control |
|---|---|
| 0 | Stops conversion operation<br>[When read]<br>Conversion stopped/standby status |
| 1 | Enables conversion operation<br>[When read]<br>While in the software trigger mode: Conversion operation status<br>While in the hardware trigger wait mode: A/D power supply stabilization wait status + conversion operation status |

<R>

| ADMD | Specification of the A/D conversion channel selection mode |
|---|---|
| 0 | Select mode |
| 1 | Scan mode |

| ADCE | A/D voltage comparator operation control Note 2 |
|---|---|
| 0 | Stops A/D voltage comparator operation |
| 1 | Enables A/D voltage comparator operation |

**Notes 1.** For details of the FR2 to FR0, LV1, LV0 bits, and A/D conversion, see **Table 11-3 A/D Conversion Time Selection**.

**2.** While in the software trigger mode or hardware trigger no-wait mode, the operation of the A/D voltage comparator is controlled by the ADCS and ADCE bits, and it takes stabilization wait status from the start of operation for the operation to stabilize. Therefore, when the ADCS bit is set to 1 after stabilization wait status or more has elapsed from the time ADCE bit is set to 1, the conversion result at that time has priority over the first conversion result. If the ADCS bit was set to 1 before the stabilization time elapsed, ignore the first conversion data.

[Stabilization wait status]

If a high-accuracy channel is selected as the analog input channel: 0.5 $\mu$s

If a test mode setting (ADTES1 bit of ADTES register = 1) is selected: 0.5 $\mu$s

If a standard channel is selected as the analog input channel: 2 $\mu$s

If a temperature sensor output/internal reference voltage output are selected as the analog input channel: (ADISS bit of ADS register = 1): 2 $\mu$s

**Cautions 1.** Change the ADMD, FR2 to FR0, LV1, and LV0 bits while in the conversion stopped status (ADCS = 0, ADCE = 0).

**2.** Setting ADCS = 1, ADCE = 0 is prohibited.

**3.** Do not change the ADCS and ADCE bits from 0 to 1 at the same time by using an 8-bit manipulation instruction. Be sure to set these bits in the order described in **11.7 A/D Converter Setup Flowchart**.

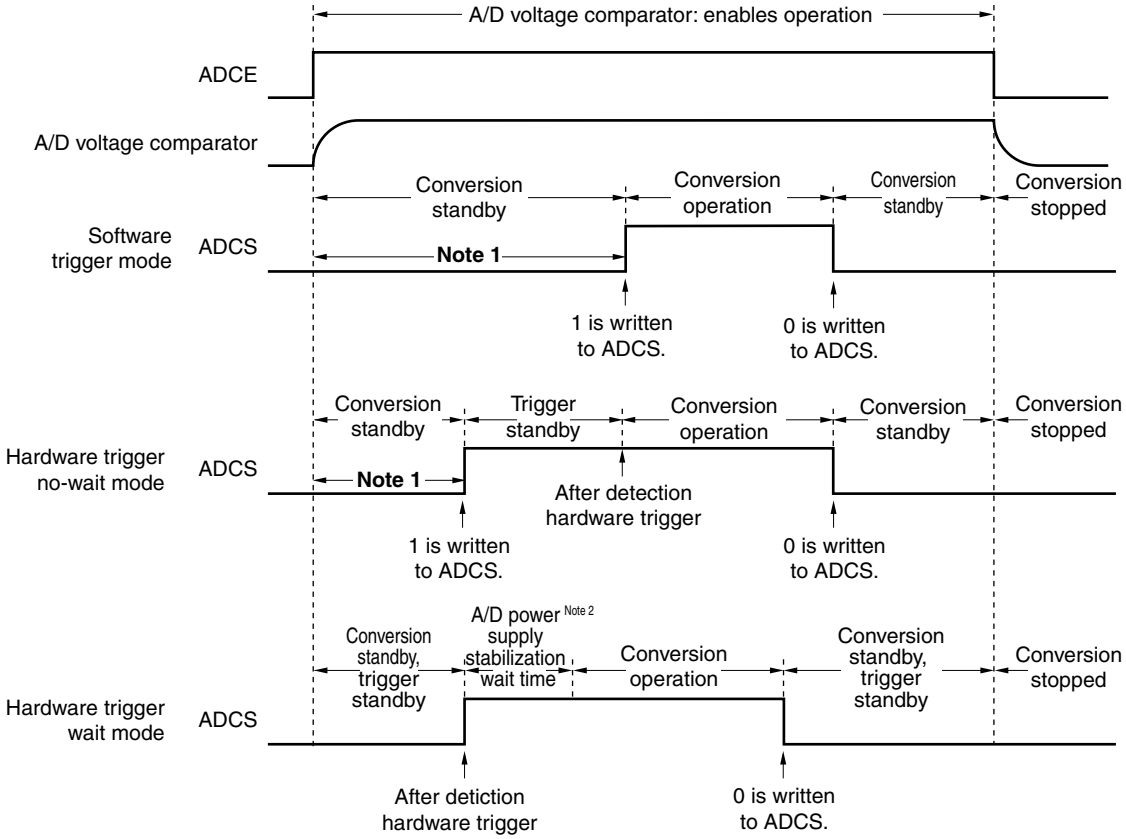
**Table 11-1. Settings of ADCS and ADCE Bits**

| ADCS | ADCE | A/D Conversion Operation |
|---|---|---|
| 0 | 0 | Conversion stopped state |
| 0 | 1 | Conversion standby state |
| 1 | 0 | Setting prohibited |
| 1 | 1 | Conversion-in-progress state |

**Table 11-2. Setting and Clearing Conditions for ADCS Bit**

| A/D Conversion Mode | | | Set Conditions | Clear Conditions |
|---|---|---|---|---|
| Software trigger | Select mode | Sequential conversion mode | When 1 is written to ADCE and ADCS | When 0 is written to ADCS |
| | | One-shot conversion mode | | • When 0 is written to ADCS<br>• The bit is automatically cleared to 0 when A/D conversion ends. |
| | Scan mode | Sequential conversion mode | | When 0 is written to ADCS |
| | | One-shot conversion mode | | • When 0 is written to ADCS<br>• The bit is automatically cleared to 0 when conversion ends on the specified four channels. |
| Hardware trigger no-wait mode | Select mode | Sequential conversion mode | | When 0 is written to ADCS |
| | | One-shot conversion mode | | When 0 is written to ADCS |
| | Scan mode | Sequential conversion mode | | When 0 is written to ADCS |
| | | One-shot conversion mode | | When 0 is written to ADCS |
| Hardware trigger wait mode | Select mode | Sequential conversion mode | When 1 is written to ADCE and a hardware trigger is input | When 0 is written to ADCS |
| | | One-shot conversion mode | | • When 0 is written to ADCS<br>• The bit is automatically cleared to 0 when A/D conversion ends. |
| | Scan mode | Sequential conversion mode | | When 0 is written to ADCS |
| | | One-shot conversion mode | | • When 0 is written to ADCS<br>• The bit is automatically cleared to 0 when conversion ends on the specified four channels. |

**Figure 11-4. Timing Chart When A/D Voltage Comparator Is Used**




**Notes 1.** While in the software trigger mode or hardware trigger no-wait mode, the time from the rising of the ADCE bit to the falling of the ADCS bit must be following time or longer to stabilize the internal circuit.

[Stabilization wait status]

If a high-accuracy channel is selected as the analog input channel: 0.5 $\mu$s

If a test mode setting (ADTES1 bit of ADTES register = 1) is selected: 0.5 $\mu$s

If a standard channel is selected as the analog input channel: 2 $\mu$s

If a temperature sensor output/internal reference voltage output are selected as the analog input channel: (ADISS bit of ADS register = 1): 2 $\mu$s

**2.** For the second and subsequent conversion in sequential conversion mode and for conversion of the channel specified by scan 1, 2, and 3 in scan mode, the A/D power supply stabilization wait time do not occur after a hardware trigger is detected.

**Cautions 1. If using the hardware trigger wait mode, setting the ADCS bit to 1 is prohibited (but the bit is automatically switched to 1 when the hardware trigger signal is detected). However, it is possible to clear the ADCS bit to 0 to specify the A/D conversion standby status.**

**2. While in the one-shot conversion mode of the hardware trigger no-wait mode, the ADCS flag is not automatically cleared to 0 when A/D conversion ends. Instead, 1 is retained.**

**3 Only rewrite the value of the ADCE bit when ADCS = 0 (while in the conversion stopped/conversion standby status).**

<R> **4. To complete A/D conversion, specify at least the following time as the hardware trigger interval:**

**Hardware trigger no wait mode: 2 $f_{CLK}$ clock + A/D conversion time**

**Hardware trigger wait mode: 2 $f_{CLK}$ clock + A/D power supply stabilization wait time + A/D conversion time**

**Remark** $f_{CLK}$: CPU/peripheral hardware clock frequency

<R>

**Table 11-3. A/D Conversion Time Selection (1/4)**

(1) 12-bit resolution mode (ADTYP = 0) When there is no stabilization wait time

(software trigger mode/hardware trigger no-wait mode)

| A/D Converter Mode Register 0 (ADM0) | | | | | Mode | Conversion Clock ($f_{AD}$) | Number of Conversion Clock | Conversion Time | Conversion Time Selection | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | $AV_{DD}$ = 1.6 to 3.6 V | $AV_{DD}$ = 1.6 to 3.6 V | $AV_{DD}$ = 1.8 to 3.6 V | $AV_{DD}$ = 2.4 to 3.6 V | $AV_{DD}$ = 2.7 to 3.6 V |
| FR2 | FR1 | FR0 | LV1 | LV0 | | | | | $f_{CLK}$ = 1 MHz | $f_{CLK}$ = 4 MHz | $f_{CLK}$ = 8 MHz | $f_{CLK}$ = 16 MHz | $f_{CLK}$ = 32 MHz |
| 0 | 0 | 0 | 0 | 0 | Normal 1 | $f_{CLK}$/32 | 54 $f_{AD}$ (number of sampling clock: 11 $f_{AD}$) | 1728/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | 54 $\mu$s[Note] |
| 0 | 0 | 1 | | | | $f_{CLK}$/16 | | 864/$f_{CLK}$ | | | | 54 $\mu$s[Note] | 27 $\mu$s[Note] |
| 0 | 1 | 0 | | | | $f_{CLK}$/8 | | 432/$f_{CLK}$ | | | 54 $\mu$s[Note] | 27 $\mu$s[Note] | 13.5 $\mu$s[Note] |
| 0 | 1 | 1 | | | | $f_{CLK}$/6 | | 324/$f_{CLK}$ | | | 40.5 $\mu$s[Note] | 20.25 $\mu$s[Note] | 10.125 $\mu$s[Note] |
| 1 | 0 | 0 | | | | $f_{CLK}$/5 | | 270/$f_{CLK}$ | | | 33.75 $\mu$s[Note] | 16.875 $\mu$s[Note] | 8.4375 $\mu$s[Note] |
| 1 | 0 | 1 | | | | $f_{CLK}$/4 | | 216/$f_{CLK}$ | | 54 $\mu$s[Note] | 27 $\mu$s[Note] | 13.5 $\mu$s[Note] | 6.75 $\mu$s[Note] |
| 1 | 1 | 0 | | | | $f_{CLK}$/2 | | 108/$f_{CLK}$ | | 27 $\mu$s[Note] | 13.5 $\mu$s[Note] | 6.75 $\mu$s[Note] | 3.375 $\mu$s[Note] |
| 1 | 1 | 1 | | | | $f_{CLK}$/1 | | 54/$f_{CLK}$ | 54 $\mu$s[Note] | 13.5 $\mu$s[Note] | 6.75 $\mu$s[Note] | 3.375 $\mu$s[Note] | Setting prohibited |
| 0 | 0 | 0 | 0 | 1 | Normal 2 | $f_{CLK}$/32 | 66 $f_{AD}$ (number of sampling clock: 23 $f_{AD}$) | 2112/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | 66 $\mu$s |
| 0 | 0 | 1 | | | | $f_{CLK}$/16 | | 1056/$f_{CLK}$ | | | | 66 $\mu$s | 33 $\mu$s |
| 0 | 1 | 0 | | | | $f_{CLK}$/8 | | 528/$f_{CLK}$ | | | 66 $\mu$s[Note] | 33 $\mu$s | 16.5 $\mu$s |
| 0 | 1 | 1 | | | | $f_{CLK}$/6 | | 396/$f_{CLK}$ | | | 49.5 $\mu$s[Note] | 24.75 $\mu$s | 12.375 $\mu$s |
| 1 | 0 | 0 | | | | $f_{CLK}$/5 | | 330/$f_{CLK}$ | | | 41.25 $\mu$s[Note] | 20.625 $\mu$s | 10.3125 $\mu$s |
| 1 | 0 | 1 | | | | $f_{CLK}$/4 | | 264/$f_{CLK}$ | | 66 $\mu$s[Note] | 33 $\mu$s[Note] | 16.5 $\mu$s | 8.25 $\mu$s |
| 1 | 1 | 0 | | | | $f_{CLK}$/2 | | 132/$f_{CLK}$ | | 33 $\mu$s[Note] | 16.5 $\mu$s[Note] | 8.25 $\mu$s | 4.125 $\mu$s |
| 1 | 1 | 1 | | | | $f_{CLK}$/1 | | 66/$f_{CLK}$ | 66 $\mu$s[Note] | 16.5 $\mu$s[Note] | 8.25 $\mu$s[Note] | 4.125 $\mu$s | Setting prohibited |
| 0 | 0 | 0 | 1 | 0 | Low-voltage 1 | $f_{CLK}$/32 | 76 $f_{AD}$ (number of sampling clock: 33 $f_{AD}$) | 2432/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | 76 $\mu$s |
| 0 | 0 | 1 | | | | $f_{CLK}$/16 | | 1216/$f_{CLK}$ | | | | 76 $\mu$s | 38 $\mu$s |
| 0 | 1 | 0 | | | | $f_{CLK}$/8 | | 608/$f_{CLK}$ | | | 76 $\mu$s | 38 $\mu$s | 19 $\mu$s |
| 0 | 1 | 1 | | | | $f_{CLK}$/6 | | 456/$f_{CLK}$ | | | 57 $\mu$s | 28.5 $\mu$s | 14.25 $\mu$s |
| 1 | 0 | 0 | | | | $f_{CLK}$/5 | | 380/$f_{CLK}$ | | | 47.5 $\mu$s | 23.75 $\mu$s | 11.875 $\mu$s |
| 1 | 0 | 1 | | | | $f_{CLK}$/4 | | 304/$f_{CLK}$ | | 76 $\mu$s[Note] | 38 $\mu$s | 19 $\mu$s | 9.5 $\mu$s |
| 1 | 1 | 0 | | | | $f_{CLK}$/2 | | 152/$f_{CLK}$ | | 38 $\mu$s[Note] | 19 $\mu$s | 9.5 $\mu$s | 4.75 $\mu$s |
| 1 | 1 | 1 | | | | $f_{CLK}$/1 | | 76/$f_{CLK}$ | 76 $\mu$s[Note] | 19 $\mu$s[Note] | 9.5 $\mu$s | 4.75 $\mu$s | Setting prohibited |
| 0 | 0 | 0 | 1 | 1 | Low-voltage 2 | $f_{CLK}$/32 | 230 $f_{AD}$ (number of sampling clock: 187 $f_{AD}$) | 7360/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | 230 $\mu$s |
| 0 | 0 | 1 | | | | $f_{CLK}$/16 | | 3680/$f_{CLK}$ | | | | 230 $\mu$s | 115 $\mu$s |
| 0 | 1 | 0 | | | | $f_{CLK}$/8 | | 1840/$f_{CLK}$ | | | 230 $\mu$s | 115 $\mu$s | 57.5 $\mu$s |
| 0 | 1 | 1 | | | | $f_{CLK}$/6 | | 1380/$f_{CLK}$ | | | 172.5 $\mu$s | 86.25 $\mu$s | 43.125 $\mu$s |
| 1 | 0 | 0 | | | | $f_{CLK}$/5 | | 1150/$f_{CLK}$ | | | 143.75 $\mu$s | 71.875 $\mu$s | 35.9375 $\mu$s |
| 1 | 0 | 1 | | | | $f_{CLK}$/4 | | 920/$f_{CLK}$ | | 230 $\mu$s | 115 $\mu$s | 57.5 $\mu$s | 28.75 $\mu$s |
| 1 | 1 | 0 | | | | $f_{CLK}$/2 | | 460/$f_{CLK}$ | | 115 $\mu$s | 57.5 $\mu$s | 28.75 $\mu$s | 14.375 $\mu$s |
| 1 | 1 | 1 | | | | $f_{CLK}$/1 | | 230/$f_{CLK}$ | 230 $\mu$s | 57.5 $\mu$s | 28.75 $\mu$s | 14.375 $\mu$s | Setting prohibited |

**Note** When using ANI16 to ANI30, setting this value is prohibited.

**(Cautions and Remark are listed on the next page.)**

**Cautions 1. The A/D conversion time must also be within the relevant range of conversion times ($t_{CONV}$) described in 29.6.1 or 30.6.1 A/D converter characteristics.**

**2. When rewriting the FR2 to FR0, LV1, and LV0 bits to other than the same data, while in the conversion stopped (ADCS = 0, ADCE = 0).**

**3. The above conversion time does not include clock frequency errors. Select conversion time, taking clock frequency errors into consideration.**

**4. When software trigger mode/hardware trigger no-wait mode, specify the conversion time so that the following conditions are satisfied:**

- **$f_{AD}$ is used within a range of 1 to 16 MHz.**
- **When using ANI16 to ANI30, the A/D converter is used in the following range of $AV_{DD}$, in accordance with the settings of the LV1 and LV0 bits:**

  **When LV1 = 0, LV0 = 0: Setting prohibit**

  **When LV1 = 0, LV0 = 1: 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V**

  **When LV1 = 1, LV0 = 0: 1.8 V ≤ $AV_{DD}$ ≤ 3.6 V**

  **When LV1 = 1, LV0 = 1: 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V**
- **Condition when temperature sensor or internal reference voltage (ADISS bit of ADS register = 1) is set for the analog input channel:**

  **When LV1 = 0, LV0 = 0: Setting prohibit**

  **When other than LV1 = 0, LV0 = 0: 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V**

**Remark** $f_{CLK}$: CPU/peripheral hardware clock frequency

<R>

**Table 11-3. A/D Conversion Time Selection (2/4)**

(2) 12-bit resolution mode (ADTYP = 0) When there is A/D power supply stabilization wait time

(hardware trigger wait mode (except second and subsequent conversion in sequential conversion mode and conversion of channel specified by scan 1, 2, and 3 in scan mode[Note 1]))

| A/D Converter Mode Register 0 (ADM0) | | | | | Mode | Conversion Clock ($f_{AD}$) | Number of A/D Power Supply Stabilization Wait Clock | Number of Conversion Clock | A/D Power Supply Stabilization Wait Time +Conversion Time | A/D Power Supply Stabilization Wait Time + Conversion Time Selection | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | $AV_{DD}$ = 1.6 to 3.6 V | $AV_{DD}$ = 1.6 to 3.6 V | $AV_{DD}$ = 1.8 to 3.6 V | $AV_{DD}$ = 2.4 to 3.6 V | $AV_{DD}$ = 2.7 to 3.6 V |
| FR2 | FR1 | FR0 | LV1 | LV0 | | | | | | $f_{CLK}$ = 1 MHz | $f_{CLK}$ = 4 MHz | $f_{CLK}$ = 8 MHz | $f_{CLK}$ = 16 MHz | $f_{CLK}$ = 32 MHz |
| 0 | 0 | 0 | 0 | 0 | Normal 1 | $f_{CLK}$/32 | 4 $f_{CLK}$ | 54 $f_{AD}$ (number of sampling clock: 11 $f_{AD}$) | 1732/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | 54.125 $\mu$s[Note 2] |
| 0 | 0 | 1 | | | | $f_{CLK}$/16 | | | 868/$f_{CLK}$ | | | | 54.25 $\mu$s[Note 2] | 27.125 $\mu$s[Note 2] |
| 0 | 1 | 0 | | | | $f_{CLK}$/8 | | | 436/$f_{CLK}$ | | | 54.5 $\mu$s[Note 2] | 27.25 $\mu$s[Note 2] | 13.625 $\mu$s[Note 2] |
| 0 | 1 | 1 | | | | $f_{CLK}$/6 | | | 328/$f_{CLK}$ | | | 41 $\mu$s[Note 2] | 20.5 $\mu$s[Note 2] | 10.25 $\mu$s[Note 2] |
| 1 | 0 | 0 | | | | $f_{CLK}$/5 | | | 274/$f_{CLK}$ | | | 34.25 $\mu$s[Note 2] | 17.125 $\mu$s[Note 2] | 8.5625 $\mu$s[Note 2] |
| 1 | 0 | 1 | | | | $f_{CLK}$/4 | | | 220/$f_{CLK}$ | | 55 $\mu$s[Note 2] | 27.5 $\mu$s[Note 2] | 13.75 $\mu$s[Note 2] | 6.875 $\mu$s[Note 2] |
| 1 | 1 | 0 | | | | $f_{CLK}$/2 | | | 112/$f_{CLK}$ | | 28 $\mu$s[Note 2] | 14 $\mu$s[Note 2] | 7 $\mu$s[Note 2] | 3.5 $\mu$s[Note 2] |
| 1 | 1 | 1 | | | | $f_{CLK}$/1 | 2 $f_{CLK}$ | | 56/$f_{CLK}$ | 56 $\mu$s[Note 2] | 14 $\mu$s[Note 2] | 7 $\mu$s[Note 2] | 3.5 $\mu$s[Note 2] | Setting prohibited |
| 0 | 0 | 0 | 0 | 1 | Normal 2 | $f_{CLK}$/32 | 58 $f_{CLK}$ | 66 $f_{AD}$ (number of sampling clock: 23 $f_{AD}$) | 2170/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | 67.8125 $\mu$s |
| 0 | 0 | 1 | | | | $f_{CLK}$/16 | | | 1114/$f_{CLK}$ | | | | 69.625 $\mu$s | 34.8125 $\mu$s |
| 0 | 1 | 0 | | | | $f_{CLK}$/8 | | | 586/$f_{CLK}$ | | | 73.25 $\mu$s[Note 2] | 36.625 $\mu$s | 18.3125 $\mu$s |
| 0 | 1 | 1 | | | | $f_{CLK}$/6 | | | 454/$f_{CLK}$ | | | 56.75 $\mu$s[Note 2] | 28.375 $\mu$s | 14.1875 $\mu$s |
| 1 | 0 | 0 | | | | $f_{CLK}$/5 | | | 388/$f_{CLK}$ | | | 48.5 $\mu$s[Note 2] | 24.25 $\mu$s | 12.125 $\mu$s |
| 1 | 0 | 1 | | | | $f_{CLK}$/4 | | | 322/$f_{CLK}$ | | 80.5 $\mu$s[Note 2] | 40.25 $\mu$s[Note 2] | 20.125 $\mu$s | 10.0625 $\mu$s |
| 1 | 1 | 0 | | | | $f_{CLK}$/2 | | | 190/$f_{CLK}$ | | 47.5 $\mu$s[Note 2] | 23.75 $\mu$s[Note 2] | 11.875 $\mu$s | 5.9375 $\mu$s |
| 1 | 1 | 1 | | | | $f_{CLK}$/1 | 29 $f_{CLK}$ | | 95/$f_{CLK}$ | 95 $\mu$s[Note 2] | 23.75 $\mu$s[Note 2] | 11.875 $\mu$s[Note 2] | 5.9375 $\mu$s | Setting prohibited |
| 0 | 0 | 0 | 1 | 0 | Low-voltage 1 | $f_{CLK}$/32 | 15 $f_{CLK}$ | 76 $f_{AD}$ (number of sampling clock: 33 $f_{AD}$) | 2447/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | 76.46875 $\mu$s[Note 2] |
| 0 | 0 | 1 | | | | $f_{CLK}$/16 | | | 1231/$f_{CLK}$ | | | | 76.9375 $\mu$s[Note 2] | 38.46875 $\mu$s[Note 2] |
| 0 | 1 | 0 | | | | $f_{CLK}$/8 | | | 623/$f_{CLK}$ | | | 77.875 $\mu$s | 38.9375 $\mu$s[Note 2] | 19.46875 $\mu$s[Note 2] |
| 0 | 1 | 1 | | | | $f_{CLK}$/6 | | | 471/$f_{CLK}$ | | | 58.875 $\mu$s | 29.4375 $\mu$s[Note 2] | 14.71875 $\mu$s[Note 2] |
| 1 | 0 | 0 | | | | $f_{CLK}$/5 | | | 395/$f_{CLK}$ | | | 49.375 $\mu$s | 24.6875 $\mu$s[Note 2] | 12.34375 $\mu$s[Note 2] |
| 1 | 0 | 1 | | | | $f_{CLK}$/4 | | | 319/$f_{CLK}$ | | 79.75 $\mu$s[Note 2] | 39.875 $\mu$s | 19.9375 $\mu$s[Note 2] | 9.96875 $\mu$s[Note 2] |
| 1 | 1 | 0 | | | | $f_{CLK}$/2 | | | 167/$f_{CLK}$ | | 41.75 $\mu$s[Note 2] | 20.875 $\mu$s | 10.4375 $\mu$s[Note 2] | 5.21875 $\mu$s[Note 2] |
| 1 | 1 | 1 | | | | $f_{CLK}$/1 | | | 91/$f_{CLK}$ | 91 $\mu$s[Note 2] | 22.75 $\mu$s[Note 2] | 11.375 $\mu$s | 5.6875 $\mu$s[Note] | Setting prohibited |
| 0 | 0 | 0 | 1 | 1 | Low-voltage 2 | $f_{CLK}$/32 | 8 $f_{CLK}$ | 230 $f_{AD}$ (number of sampling clock: 187 $f_{AD}$) | 7368/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | 230.25 $\mu$s[Note 2] |
| 0 | 0 | 1 | | | | $f_{CLK}$/16 | | | 3688/$f_{CLK}$ | | | | 230.5 $\mu$s[Note 2] | 115.25 $\mu$s[Note 2] |
| 0 | 1 | 0 | | | | $f_{CLK}$/8 | | | 1848/$f_{CLK}$ | | | 231 $\mu$s[Note 2] | 115.5 $\mu$s[Note 2] | 57.75 $\mu$s[Note 2] |
| 0 | 1 | 1 | | | | $f_{CLK}$/6 | | | 1388/$f_{CLK}$ | | | 173.5 $\mu$s[Note 2] | 86.75 $\mu$s[Note 2] | 43.375 $\mu$s[Note 2] |
| 1 | 0 | 0 | | | | $f_{CLK}$/5 | | | 1158/$f_{CLK}$ | | | 144.75 $\mu$s[Note 2] | 72.375 $\mu$s[Note 2] | 36.1875 $\mu$s[Note 2] |
| 1 | 0 | 1 | | | | $f_{CLK}$/4 | | | 928/$f_{CLK}$ | | 232 $\mu$s | 116 $\mu$s[Note 2] | 58 $\mu$s[Note 2] | 29 $\mu$s[Note 2] |
| 1 | 1 | 0 | | | | $f_{CLK}$/2 | | | 468/$f_{CLK}$ | | 117 $\mu$s | 58.5 $\mu$s[Note 2] | 29.25 $\mu$s[Note 2] | 14.625 $\mu$s[Note 2] |
| 1 | 1 | 1 | | | | $f_{CLK}$/1 | | | 238/$f_{CLK}$ | 238 $\mu$s | 59.5 $\mu$s | 29.75 $\mu$s[Note 2] | 14.875 $\mu$s[Note 2] | Setting prohibited |

**(Notes, Cautions and Remark are listed on the next page.)**

**Notes 1.** For the second and subsequent conversion in sequential conversion mode and for conversion of the channel specified by scan 1, 2, and 3 in scan mode, the conversion start time and stabilization wait time for A/D power supply do not occur after a hardware trigger is detected (see **Table 11-3 (1/4)**).

**2.** When using ANI16 to ANI30, setting this value is prohibited.

**Cautions 1. The A/D conversion time must also be within the relevant range of conversion times ($t_{CONV}$) described in 29.6.1 or 30.6.1 A/D converter characteristics.**

**2. When rewriting the FR2 to FR0, LV1, and LV0 bits to other than the same data, while in the conversion stopped (ADCS = 0, ADCE = 0).**

**3. The above conversion time does not include clock frequency errors. Select conversion time, taking clock frequency errors into consideration.**

**4. When hardware trigger wait mode, specify the conversion time so that the following conditions are satisfied:**

- **$f_{AD}$ is used within a range of 1 to 16 MHz.**
- **When using ANI16 to ANI30, the A/D converter is used in the following conditions:**

  **When LV1 = 0, LV0 = 0: Setting prohibit**

  **When LV1 = 0, LV0 = 1: 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V**

  **When LV1 = 1, LV0 = 0: 1.8 V ≤ $AV_{DD}$ ≤ 3.6 V, 1 MHz ≤ $f_{CLK}$ ≤ 8 MHz**

  **When LV1 = 1, LV0 = 1: 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V, 1 MHz ≤ $f_{CLK}$ ≤ 4 MHz**
- **Condition when temperature sensor or internal reference voltage (ADISS bit of ADS register = 1) is set for the analog input channel:**

  **When LV1 = 0, LV0 = 0: Setting prohibit**

  **When LV1 = 0, LV0 = 1: 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V**

  **When LV1 = 1, LV0 = 0: 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V, 1 MHz ≤ $f_{CLK}$ ≤ 8 MHz**

  **When LV1 = 1, LV0 = 1: 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V, 1 MHz ≤ $f_{CLK}$ ≤ 4 MHz**

**Remark** $f_{CLK}$: CPU/peripheral hardware clock frequency

<R>

**Table 11-3. A/D Conversion Time Selection (3/4)**

(3) 8-bit resolution mode (ADTYP = 1) When there is no stabilization wait time

(software trigger mode/hardware trigger no-wait mode)

| A/D Converter Mode Register 0 (ADM0) | | | | | Mode | Conversion Clock ($f_{AD}$) | Number of Conversion Clock | Conversion Time | Conversion Time Selection | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | $AV_{DD}$ = 1.6 to 3.6 V | $AV_{DD}$ = 1.6 to 3.6 V | $AV_{DD}$ = 1.8 to 3.6 V | $AV_{DD}$ = 2.4 to 3.6 V | $AV_{DD}$ = 2.7 to 3.6 V |
| FR2 | FR1 | FR0 | LV1 | LV0 | | | | | $f_{CLK}$ = 1 MHz | $f_{CLK}$ = 4 MHz | $f_{CLK}$ = 8 MHz | $f_{CLK}$ = 16 MHz | $f_{CLK}$ = 32 MHz |
| 0 | 0 | 0 | 0 | 0 | Normal 1 | $f_{CLK}$/32 | 41 $f_{AD}$ (number of sampling clock: 11 $f_{AD}$) | 1312/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | 41 $\mu$s[Note] |
| 0 | 0 | 1 | | | | $f_{CLK}$/16 | | 656/$f_{CLK}$ | | | | 41 $\mu$s[Note] | 20.5 $\mu$s[Note] |
| 0 | 1 | 0 | | | | $f_{CLK}$/8 | | 328/$f_{CLK}$ | | | 41 $\mu$s[Note] | 20.5 $\mu$s[Note] | 10.25 $\mu$s[Note] |
| 0 | 1 | 1 | | | | $f_{CLK}$/6 | | 246/$f_{CLK}$ | | | 30.75 $\mu$s[Note] | 15.375 $\mu$s[Note] | 7.6875 $\mu$s[Note] |
| 1 | 0 | 0 | | | | $f_{CLK}$/5 | | 205/$f_{CLK}$ | | | 25.625 $\mu$s[Note] | 12.8125 $\mu$s[Note] | 6.40625 $\mu$s[Note] |
| 1 | 0 | 1 | | | | $f_{CLK}$/4 | | 164/$f_{CLK}$ | | 41 $\mu$s[Note] | 20.5 $\mu$s[Note] | 10.25 $\mu$s[Note] | 5.125 $\mu$s[Note] |
| 1 | 1 | 0 | | | | $f_{CLK}$/2 | | 82/$f_{CLK}$ | | 20.5 $\mu$s[Note] | 10.25 $\mu$s[Note] | 5.125 $\mu$s[Note] | 2.5625 $\mu$s[Note] |
| 1 | 1 | 1 | | | | $f_{CLK}$/1 | | 41/$f_{CLK}$ | 41 $\mu$s[Note] | 10.25 $\mu$s[Note] | 5.125 $\mu$s[Note] | 2.5625 $\mu$s[Note] | Setting prohibited |
| 0 | 0 | 0 | 0 | 1 | Normal 2 | $f_{CLK}$/32 | 53 $f_{AD}$ (number of sampling clock: 23 $f_{AD}$) | 1696/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | 53 $\mu$s |
| 0 | 0 | 1 | | | | $f_{CLK}$/16 | | 848/$f_{CLK}$ | | | | 53 $\mu$s | 26.5 $\mu$s |
| 0 | 1 | 0 | | | | $f_{CLK}$/8 | | 424/$f_{CLK}$ | | | 53 $\mu$s[Note] | 26.5 $\mu$s | 13.25 $\mu$s |
| 0 | 1 | 1 | | | | $f_{CLK}$/6 | | 318/$f_{CLK}$ | | | 39.75 $\mu$s[Note] | 19.875 $\mu$s | 9.9375 $\mu$s |
| 1 | 0 | 0 | | | | $f_{CLK}$/5 | | 265/$f_{CLK}$ | | | 33.125 $\mu$s[Note] | 16.5625 $\mu$s | 8.28125 $\mu$s |
| 1 | 0 | 1 | | | | $f_{CLK}$/4 | | 212/$f_{CLK}$ | | 53 $\mu$s[Note] | 26.5 $\mu$s[Note] | 13.25 $\mu$s | 6.625 $\mu$s |
| 1 | 1 | 0 | | | | $f_{CLK}$/2 | | 106/$f_{CLK}$ | | 26.5 $\mu$s[Note] | 13.25 $\mu$s[Note] | 6.625 $\mu$s | 3.3125 $\mu$s |
| 1 | 1 | 1 | | | | $f_{CLK}$/1 | | 53/$f_{CLK}$ | 53 $\mu$s[Note] | 13.25 $\mu$s[Note] | 6.625 $\mu$s[Note] | 3.3125 $\mu$s | Setting prohibited |
| 0 | 0 | 0 | 1 | 0 | Low-voltage 1 | $f_{CLK}$/32 | 63 $f_{AD}$ (number of sampling clock: 33 $f_{AD}$) | 2016/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | 63 $\mu$s |
| 0 | 0 | 1 | | | | $f_{CLK}$/16 | | 1008/$f_{CLK}$ | | | | 63 $\mu$s | 31.5 $\mu$s |
| 0 | 1 | 0 | | | | $f_{CLK}$/8 | | 504/$f_{CLK}$ | | | 63 $\mu$s | 31.5 $\mu$s | 15.75 $\mu$s |
| 0 | 1 | 1 | | | | $f_{CLK}$/6 | | 378/$f_{CLK}$ | | | 47.25 $\mu$s | 23.625 $\mu$s | 11.8125 $\mu$s |
| 1 | 0 | 0 | | | | $f_{CLK}$/5 | | 315/$f_{CLK}$ | | | 39.375 $\mu$s | 19.6875 $\mu$s | 9.84375 $\mu$s |
| 1 | 0 | 1 | | | | $f_{CLK}$/4 | | 252/$f_{CLK}$ | | 63 $\mu$s[Note] | 31.5 $\mu$s | 15.75 $\mu$s | 7.875 $\mu$s |
| 1 | 1 | 0 | | | | $f_{CLK}$/2 | | 126/$f_{CLK}$ | | 31.5 $\mu$s[Note] | 15.75 $\mu$s | 7.875 $\mu$s | 3.9375 $\mu$s |
| 1 | 1 | 1 | | | | $f_{CLK}$/1 | | 63/$f_{CLK}$ | 63 $\mu$s[Note] | 15.75 $\mu$s[Note] | 7.875 $\mu$s | 3.9375 $\mu$s | Setting prohibited |
| 0 | 0 | 0 | 1 | 1 | Low-voltage 2 | $f_{CLK}$/32 | 217 $f_{AD}$ (number of sampling clock: 187 $f_{AD}$) | 6944/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | 217 $\mu$s |
| 0 | 0 | 1 | | | | $f_{CLK}$/16 | | 3472/$f_{CLK}$ | | | | 217 $\mu$s | 108.5 $\mu$s |
| 0 | 1 | 0 | | | | $f_{CLK}$/8 | | 1736/$f_{CLK}$ | | | 217 $\mu$s | 108.5 $\mu$s | 54.25 $\mu$s |
| 0 | 1 | 1 | | | | $f_{CLK}$/6 | | 1302/$f_{CLK}$ | | | 162.75 $\mu$s | 81.375 $\mu$s | 40.6875 $\mu$s |
| 1 | 0 | 0 | | | | $f_{CLK}$/5 | | 1085/$f_{CLK}$ | | | 135.625 $\mu$s | 67.8125 $\mu$s | 33.90625 $\mu$s |
| 1 | 0 | 1 | | | | $f_{CLK}$/4 | | 868/$f_{CLK}$ | | 217 $\mu$s | 108.5 $\mu$s | 54.25 $\mu$s | 27.125 $\mu$s |
| 1 | 1 | 0 | | | | $f_{CLK}$/2 | | 434/$f_{CLK}$ | | 108.5 $\mu$s | 54.25 $\mu$s | 27.125 $\mu$s | 13.5625 $\mu$s |
| 1 | 1 | 1 | | | | $f_{CLK}$/1 | | 217/$f_{CLK}$ | 217 $\mu$s | 54.25 $\mu$s | 27.125 $\mu$s | 13.5625 $\mu$s | Setting prohibited |

**Note** When using ANI16 to ANI30, setting this value is prohibited.

**(Cautions and Remark are listed on the next page.)**

**Cautions 1. The A/D conversion time must also be within the relevant range of conversion times ($t_{CONV}$) described in 29.6.1 or 30.6.1 A/D converter characteristics.**

**2. When rewriting the FR2 to FR0, LV1, and LV0 bits to other than the same data, while in the conversion stopped (ADCS = 0, ADCE = 0).**

**3. The above conversion time does not include clock frequency errors. Select conversion time, taking clock frequency errors into consideration.**

**4. When software trigger mode/hardware trigger no-wait mode, specify the conversion time so that the following conditions are satisfied:**

- **$f_{AD}$ is used within a range of 1 to 16 MHz.**
- **When using ANI16 to ANI30, the A/D converter is used in the following range of $AV_{DD}$, in accordance with the settings of the LV1 and LV0 bits:**

  **When LV1 = 0, LV0 = 0: Setting prohibit**

  **When LV1 = 0, LV0 = 1: 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V**

  **When LV1 = 1, LV0 = 0: 1.8 V ≤ $AV_{DD}$ ≤ 3.6 V**

  **When LV1 = 1, LV0 = 1: 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V**
- **Condition when temperature sensor or internal reference voltage (ADISS bit of ADS register = 1) is set for the analog input channel:**

  **When LV1 = 0, LV0 = 0: Setting prohibit**

  **When other than LV1 = 0, LV0 = 0: 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V**

**Remark** $f_{CLK}$: CPU/peripheral hardware clock frequency

<R>

**Table 11-3. A/D Conversion Time Selection (4/4)**

(4) 8-bit resolution mode (ADTYP = 1) When there is A/D power supply stabilization wait time
(hardware trigger wait mode (except second and subsequent conversion in sequential conversion mode and conversion of channel specified by scan 1, 2, and 3 in scan mode[Note 1]))

| A/D Converter Mode Register 0 (ADM0) | | | | | Mode | Conversion Clock ($f_{AD}$) | Number of A/D Power Supply Stabilization Wait Clock | Number of Conversion Clock | A/D Power Supply Stabilization Wait Time +Conversion Time | A/D Power Supply Stabilization Wait Time + Conversion Time Selection | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | $AV_{DD}$ = 1.6 to 3.6 V | $AV_{DD}$ = 1.6 to 3.6 V | $AV_{DD}$ = 1.8 to 3.6 V | $AV_{DD}$ = 2.4 to 3.6 V | $AV_{DD}$ = 2.7 to 3.6 V |
| FR2 | FR1 | FR0 | LV1 | LV0 | | | | | | $f_{CLK}$ = 1 MHz | $f_{CLK}$ = 4 MHz | $f_{CLK}$ = 8 MHz | $f_{CLK}$ = 16 MHz | $f_{CLK}$ = 32 MHz |
| 0 | 0 | 0 | 0 | 0 | Normal 1 | $f_{CLK}$/32 | 4 $f_{CLK}$ | 41 $f_{AD}$ (number of sampling clock: 11 $f_{AD}$) | 1316/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | 41.125 $\mu$s[Note 2] |
| 0 | 0 | 1 | | | | $f_{CLK}$/16 | | | 660/$f_{CLK}$ | | | | 41.25 $\mu$s[Note 2] | 20.625 $\mu$s[Note 2] |
| 0 | 1 | 0 | | | | $f_{CLK}$/8 | | | 332/$f_{CLK}$ | | | 41.5 $\mu$s[Note 2] | 20.75 $\mu$s[Note 2] | 10.375 $\mu$s[Note 2] |
| 0 | 1 | 1 | | | | $f_{CLK}$/6 | | | 250/$f_{CLK}$ | | | 31.25 $\mu$s[Note 2] | 15.625 $\mu$s[Note 2] | 7.8125 $\mu$s[Note 2] |
| 1 | 0 | 0 | | | | $f_{CLK}$/5 | | | 209/$f_{CLK}$ | | | 25.125 $\mu$s[Note 2] | 13.0625 $\mu$s[Note 2] | 6.53125 $\mu$s[Note 2] |
| 1 | 0 | 1 | | | | $f_{CLK}$/4 | | | 168/$f_{CLK}$ | | 42 $\mu$s[Note 2] | 21 $\mu$s[Note 2] | 10.5 $\mu$s[Note 2] | 5.25 $\mu$s[Note 2] |
| 1 | 1 | 0 | | | | $f_{CLK}$/2 | | | 86/$f_{CLK}$ | | 21.5 $\mu$s[Note 2] | 10.75 $\mu$s[Note 2] | 5.375 $\mu$s[Note 2] | 2.6875 $\mu$s[Note 2] |
| 1 | 1 | 1 | | | | $f_{CLK}$/1 | 2 $f_{CLK}$ | | 43/$f_{CLK}$ | 43 $\mu$s[Note 2] | 10.75 $\mu$s[Note 2] | 5.375 $\mu$s[Note 2] | 2.6875 $\mu$s[Note 2] | Setting prohibited |
| 0 | 0 | 0 | 0 | 1 | Normal 2 | $f_{CLK}$/32 | 58 $f_{CLK}$ | 53 $f_{AD}$ (number of sampling clock: 23 $f_{AD}$) | 1754/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | 54.8125 $\mu$s |
| 0 | 0 | 1 | | | | $f_{CLK}$/16 | | | 906/$f_{CLK}$ | | | | 56.625 $\mu$s | 28.3125 $\mu$s |
| 0 | 1 | 0 | | | | $f_{CLK}$/8 | | | 482/$f_{CLK}$ | | | 60.25 $\mu$s[Note 2] | 30.125 $\mu$s | 15.0625 $\mu$s |
| 0 | 1 | 1 | | | | $f_{CLK}$/6 | | | 376/$f_{CLK}$ | | | 47 $\mu$s[Note 2] | 23.5 $\mu$s | 11.75 $\mu$s |
| 1 | 0 | 0 | | | | $f_{CLK}$/5 | | | 323/$f_{CLK}$ | | | 40.375 $\mu$s[Note 2] | 20.1875 $\mu$s | 10.09375 $\mu$s |
| 1 | 0 | 1 | | | | $f_{CLK}$/4 | | | 270/$f_{CLK}$ | | 67.5 $\mu$s[Note 2] | 33.75 $\mu$s[Note 2] | 16.875 $\mu$s | 8.4375 $\mu$s |
| 1 | 1 | 0 | | | | $f_{CLK}$/2 | | | 164/$f_{CLK}$ | | 41 $\mu$s[Note 2] | 20.5 $\mu$s[Note 2] | 10.25 $\mu$s | 5.125 $\mu$s |
| 1 | 1 | 1 | | | | $f_{CLK}$/1 | 29 $f_{CLK}$ | | 82/$f_{CLK}$ | 82 $\mu$s[Note 2] | 20.5 $\mu$s[Note 2] | 10.25 $\mu$s[Note 2] | 5.125 $\mu$s | Setting prohibited |
| 0 | 0 | 0 | 1 | 0 | Low-voltage 1 | $f_{CLK}$/32 | 15 $f_{CLK}$ | 63 $f_{AD}$ (number of sampling clock: 33 $f_{AD}$) | 2031/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | 63.46875 $\mu$s[Note 2] |
| 0 | 0 | 1 | | | | $f_{CLK}$/16 | | | 1023/$f_{CLK}$ | | | | 63.9375 $\mu$s[Note 2] | 31.96875 $\mu$s[Note 2] |
| 0 | 1 | 0 | | | | $f_{CLK}$/8 | | | 519/$f_{CLK}$ | | | 64.875 $\mu$s | 32.4375 $\mu$s[Note 2] | 16.21875 $\mu$s[Note 2] |
| 0 | 1 | 1 | | | | $f_{CLK}$/6 | | | 393/$f_{CLK}$ | | | 49.125 $\mu$s | 24.5625 $\mu$s[Note 2] | 12.28125 $\mu$s[Note 2] |
| 1 | 0 | 0 | | | | $f_{CLK}$/5 | | | 330/$f_{CLK}$ | | | 41.25 $\mu$s | 20.625 $\mu$s[Note 2] | 10.3125 $\mu$s[Note 2] |
| 1 | 0 | 1 | | | | $f_{CLK}$/4 | | | 267/$f_{CLK}$ | | 66.75 $\mu$s[Note 2] | 33.375 $\mu$s | 16.6875 $\mu$s[Note 2] | 8.34375 $\mu$s[Note 2] |
| 1 | 1 | 0 | | | | $f_{CLK}$/2 | | | 141/$f_{CLK}$ | | 35.25 $\mu$s[Note 2] | 17.625 $\mu$s | 8.8125 $\mu$s[Note 2] | 4.40625 $\mu$s[Note 2] |
| 1 | 1 | 1 | | | | $f_{CLK}$/1 | | | 78/$f_{CLK}$ | 78 $\mu$s[Note 2] | 19.5 $\mu$s[Note 2] | 9.75 $\mu$s | 4.875 $\mu$s[Note 2] | Setting prohibited |
| 0 | 0 | 0 | 1 | 1 | Low-voltage 2 | $f_{CLK}$/32 | 8 $f_{CLK}$ | 217 $f_{AD}$ (number of sampling clock: 187 $f_{AD}$) | 6952/$f_{CLK}$ | Setting prohibited | Setting prohibited | Setting prohibited | Setting prohibited | 217.25 $\mu$s[Note 2] |
| 0 | 0 | 1 | | | | $f_{CLK}$/16 | | | 3480/$f_{CLK}$ | | | | 217.5 $\mu$s[Note 2] | 108.75 $\mu$s[Note 2] |
| 0 | 1 | 0 | | | | $f_{CLK}$/8 | | | 1744/$f_{CLK}$ | | | 218 $\mu$s | 109 $\mu$s[Note 2] | 54.5 $\mu$s[Note 2] |
| 0 | 1 | 1 | | | | $f_{CLK}$/6 | | | 1310/$f_{CLK}$ | | | 163.75 $\mu$s[Note 2] | 81.875 $\mu$s[Note 2] | 40.9375 $\mu$s[Note 2] |
| 1 | 0 | 0 | | | | $f_{CLK}$/5 | | | 1093/$f_{CLK}$ | | | 136.625 $\mu$s[Note 2] | 68.3125 $\mu$s[Note 2] | 34.15625 $\mu$s[Note 2] |
| 1 | 0 | 1 | | | | $f_{CLK}$/4 | | | 876/$f_{CLK}$ | | 219 $\mu$s | 109.5 $\mu$s[Note 2] | 54.75 $\mu$s[Note 2] | 27.375 $\mu$s[Note 2] |
| 1 | 1 | 0 | | | | $f_{CLK}$/2 | | | 442/$f_{CLK}$ | | 110.5 $\mu$s | 55.25 $\mu$s[Note 2] | 27.625 $\mu$s[Note 2] | 13.8125 $\mu$s[Note 2] |
| 1 | 1 | 1 | | | | $f_{CLK}$/1 | | | 225/$f_{CLK}$ | 225 $\mu$s | 56.25 $\mu$s | 28.125 $\mu$s[Note 2] | 14.0625 $\mu$s[Note 2] | Setting prohibited |

**(Notes, Cautions and Remark are listed on the next page.)**

**Notes 1.** For the second and subsequent conversion in sequential conversion mode and for conversion of the channel specified by scan 1, 2, and 3 in scan mode, the conversion start time and stabilization wait time for A/D power supply do not occur after a hardware trigger is detected (see **Table 11-3 (3/4)**).

**2.** When using ANI16 to ANI30, setting this value is prohibited.

**Cautions 1. The A/D conversion time must also be within the relevant range of conversion times ($t_{CONV}$) described in 29.6.1 or 30.6.1 A/D converter characteristics.**

**2. When rewriting the FR2 to FR0, LV1, and LV0 bits to other than the same data, while in the conversion stopped (ADCS = 0, ADCE = 0).**

**3. The above conversion time does not include clock frequency errors. Select conversion time, taking clock frequency errors into consideration.**

**4. When hardware trigger wait mode, specify the conversion time so that the following conditions are satisfied:**

- **$f_{AD}$ is used within a range of 1 to 16 MHz.**
- **When using ANI16 to ANI30, the A/D converter is used in the following conditions:**

  **When LV1 = 0, LV0 = 0: Setting prohibit**

  **When LV1 = 0, LV0 = 1: 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V**

  **When LV1 = 1, LV0 = 0: 1.8 V ≤ $AV_{DD}$ ≤ 3.6 V, 1 MHz ≤ $f_{CLK}$ ≤ 8 MHz**

  **When LV1 = 1, LV0 = 1: 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V, 1 MHz ≤ $f_{CLK}$ ≤ 4 MHz**
- **Condition when temperature sensor or internal reference voltage (ADISS bit of ADS register = 1) is set for the analog input channel:**

  **When LV1 = 0, LV0 = 0: Setting prohibit**

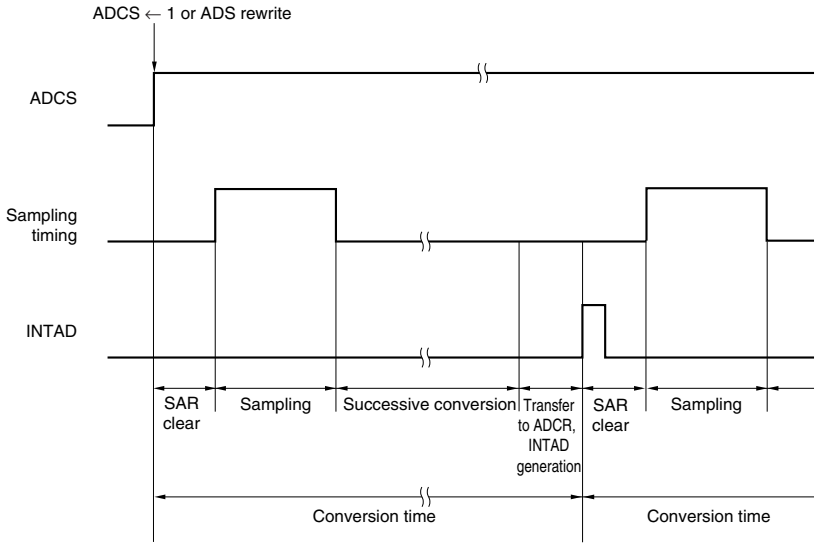
  **When LV1 = 0, LV0 = 1: 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V**

  **When LV1 = 1, LV0 = 0: 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V, 1 MHz ≤ $f_{CLK}$ ≤ 8 MHz**

  **When LV1 = 1, LV0 = 1: 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V, 1 MHz ≤ $f_{CLK}$ ≤ 4 MHz**

**Remark** $f_{CLK}$: CPU/peripheral hardware clock frequency

**Figure 11-5. A/D Converter Sampling and A/D Conversion Timing (Example for Software Trigger Mode)**



### 11.3.3 A/D converter mode register 1 (ADM1)

This register is used to specify the A/D conversion trigger, conversion mode, and hardware trigger signal.

The ADM1 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 11-6. Format of A/D Converter Mode Register 1 (ADM1)**

Address: FFF32H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADM1 | ADTMD1 | ADTMD0 | ADSCM | 0 | 0 | 0 | ADTRS1 | ADTRS0 |

| ADTMD1 | ADTMD0 | Selection of the A/D conversion trigger mode |
|---|---|---|
| 0 | × | Software trigger mode |
| 1 | 0 | Hardware trigger no-wait mode |
| 1 | 1 | Hardware trigger wait mode |

| ADSCM | Specification of the A/D conversion mode |
|---|---|
| 0 | Sequential conversion mode |
| 1 | One-shot conversion mode |

| ADTRS1 | ADTRS0 | Selection of the hardware trigger signal |
|---|---|---|
| 0 | 0 | End of timer channel 01 count or capture interrupt signal (INTTM01) |
| 0 | 1 | Setting prohibited |
| 1 | 0 | Real-time clock interrupt signal (INTRTC) |
| 1 | 1 | Interval timer interrupt signal (INTIT) |

**Cautions 1. Rewrite the value of the ADM1 register while conversion is stopped (ADCS = 0, ADCE = 0).**

**2. To complete A/D conversion, specify at least the following time as the hardware trigger interval:**

<R> **Hardware trigger no wait mode: 2 $f_{CLK}$ clock + A/D conversion time**

**Hardware trigger wait mode: 2 $f_{CLK}$ clock + A/D power supply stabilization wait time + A/D conversion time**

<R> **3. In modes other than SNOOZE mode, input of the next INTRTC or INTIT will not be recognized as a valid hardware trigger for up to four $f_{CLK}$ cycles after the first INTRTC or INTIT is input.**

**Remarks 1.** ×: don't care

**2.** $f_{CLK}$: CPU/peripheral hardware clock frequency

### 11.3.4 A/D converter mode register 2 (ADM2)

This register is used to select the + side or - side reference voltage of the A/D converter, check the upper limit and lower limit A/D conversion result values, select the resolution, and specify whether to use the SNOOZE mode.

The ADM2 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 11-7. Format of A/D Converter Mode Register 2 (ADM2) (1/2)**

Address: F0010H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | <3> | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| ADM2 | ADREFP1 | ADREFP0 | ADREFM | 0 | ADRCK | AWC | 0 | ADTYP |

| ADREFP1 | ADREFP0 | Selection of the + side reference voltage of the A/D converter |
|---|---|---|
| 0 | 0 | Supplied from $AV_{DD}$ |
| 0 | 1 | Supplied from P20/$AV_{REFP}$/ANI0 |
| 1 | 0 | Supplied from the internal reference voltage (1.45 V)Note |
| 1 | 1 | Setting prohibited |

- When ADREFP1 or ADREFP0 bit is rewritten, this must be configured in accordance with the following procedures.
  (1) Set ADCE = 0
  (2) Change the values of ADREFP1 and ADREFP0
  (3) Reference voltage stabilization wait time (A)
  (4) Set ADCE = 1
  (5) Reference voltage stabilization wait time (B)

  The stabilization wait time indicated by (3) is required when the value of the ADREFP1 and ADREFP0 bits is changed.
  When ADREFP1 and ADREFP0 are changed to 1 and 0: A = 10 $\mu$s
  When ADREFP1 and ADREFP0 are changed to 0 and 0 or 0 and 1: A = 1 $\mu$s
  The stabilization wait time indicated by (5) is required when the value of the ADCE bit is changed to 1.
  If a high-accuracy channel is selected as the analog input channel: B = 0.5 $\mu$s
  If a test mode setting (ADTES1 bit of ADTES register = 1) is selected: B = 0.5 $\mu$s
  If a standard channel is selected as the analog input channel: B = 2 $\mu$s
  If a temperature sensor/internal reference voltage are selected as the analog input channel: (ADISS bit of ADS register = 1): B = 2 $\mu$s
  After (5) stabilization time, start the A/D conversion.
- When ADREFP1 and ADREFP0 are set to 1 and 0, respectively, A/D conversion cannot be performed on the temperature sensor output voltage and internal reference voltage (1.45 V).
  Be sure to perform A/D conversion while ADISS = 0.

| ADREFM | Selection of the − side reference voltage of the A/D converter |
|---|---|
| 0 | Supplied from $AV_{SS}$ |
| 1 | Supplied from P21/$AV_{REFM}$/ANI1 |

**Note** This setting can be used only in HS (high-speed main) mode. For detail, see Figure 24-3 Format of User Option Byte (000C2H/010C2H).

**Cautions 1. Rewrite the value of the ADM2 register while conversion is stopped (ADCS = 0, ADCE = 0).**

**2. Do not set the ADREFP1 bit to 1 when shifting to STOP mode, or to HALT mode while the CPU is operating on the subsystem clock. Also, if the ADREFP1 bit is set to 1, the A/D converter reference voltage current ($I_{ADREF}$) indicated in 29.3.2 or 30.3.2 Supply current characteristics will be added to the current consumption when shifting to HALT mode while the CPU is operating on the main system clock.**

**3. When using $AV_{REFP}$ and $AV_{REFM}$, specify ANI0 and ANI1 as the analog input channels and specify input mode by using the port mode register.**

**Figure 11-7. Format of A/D Converter Mode Register 2 (ADM2) (2/2)**

Address: F0010H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | <3> | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| ADM2 | ADREFP1 | ADREFP0 | ADREFM | 0 | ADRCK | AWC | 0 | ADTYP |

| ADRCK | Checking the upper limit and lower limit conversion result values |
|---|---|
| 0 | The A/D conversion end interrupt request signal (INTAD) is output when the ADLL register ≤ the ADCR register ≤ the ADUL register (AREA1). |
| 1 | The A/D conversion end interrupt request signal (INTAD) is output when the ADCR register < the ADLL register (AREA2) or the ADUL register < the ADCR register (AREA3). |
| Figure 11-8 shows the generation range of the A/D conversion end interrupt request signal (INTAD) for AREA1 to AREA3. | |

| AWC | Specification of the SNOOZE mode |
|---|---|
| 0 | Do not use the SNOOZE mode function. |
| 1 | Use the SNOOZE mode function. |

When there is a hardware trigger signal in the STOP mode, the STOP mode is exited, and A/D conversion is performed without operating the CPU (the SNOOZE mode).

- The SNOOZE mode function can only be specified when the high-speed on-chip oscillator clock is selected for the CPU/peripheral hardware clock ($f_{CLK}$). If any other clock is selected, specifying this mode is prohibited.
- Using the SNOOZE mode function in the software trigger mode or hardware trigger no-wait mode is prohibited.
- Using the SNOOZE mode function in the sequential conversion mode is prohibited.
- When using the SNOOZE mode function, specify a hardware trigger interval of at least "shift time to SNOOZE mode[Note 1] + A/D power supply stabilization wait time + A/D conversion time +2 $f_{CLK}$ clock"
- Even when using SNOOZE mode, be sure to set the AWC bit to 0 in normal operation and change it to 1 just before shifting to STOP mode.
  Also, be sure to change the AWC bit to 0 after returning from STOP mode to normal operation.
  If the AWC bit is left set to 1, A/D conversion will not start normally in spite of the subsequent SNOOZE or normal operation.

| ADTYP | Selection of the A/D conversion resolution |
|---|---|
| 0 | 12-bit resolution[Note 2] |
| 1 | 8-bit resolution |

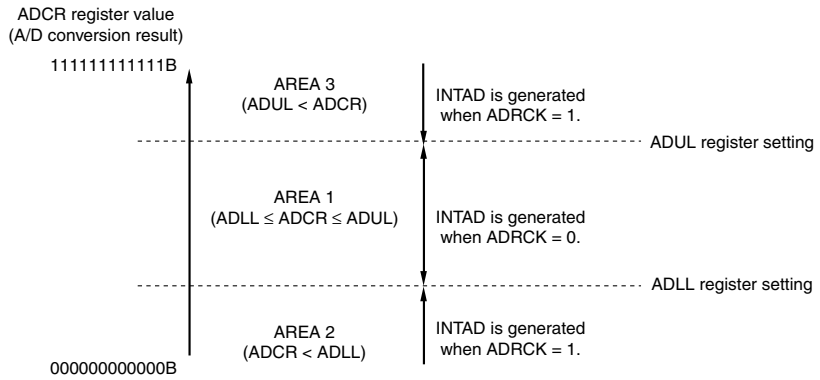
**Notes 1.** Refer to "Transition time from STOP mode to SNOOZE mode:" in **18.3.3 SNOOZE mode**

<R> **2.** The valid resolution differs depending on the voltage conditions of $AV_{DD}$ and $AV_{REFP}$.
For details, see **29.6.1 A/D converter characteristics**.

**Caution Rewrite the value of the ADM2 register while conversion operation is stopped (ADCS = 0, ADCE = 0).**

**Figure 11-8. ADRCK Bit Interrupt Signal Generation Range**

<R>




<R>

**Remark** If INTAD does not occur, the A/D conversion result is not stored in the ADCR or ADCRH register.

### 11.3.5 12-bit A/D conversion result register (ADCR)

This register is a 16-bit register that stores the A/D conversion result. The higher 4 bits are fixed to 0. Each time A/D conversion ends, each time A/D conversion ends, the value of ADSAR [11:0] is stored in the A/D conversion result register (note that whether to store this value is determined by the setting of the ADRCK bit of the ADM2 register and by the settings of the ADUL and ADLL registers). The higher 4 bits of the conversion result are stored in FFF1FH and the lower 8 bits are stored in the lower 4 bits of FFF1EH[Note].

The ADCR register can be read by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0000H.

**Note** If the A/D conversion result is outside the range specified by using the A/D conversion comparison function (the value specified by the ADRCK bit of the ADM2 register and ADUL/ADLL registers; see **Figure 11-8**), the result is not stored.

<R> **Caution The valid resolution differs depending on the voltage conditions of $AV_{DD}$ and $AV_{REFP}$. For details, see 29.6.1 A/D converter characteristics.**

<R> **Remarks 1.** When using the converter with a resolution of 10 bits, select the 12-bit resolution mode (ADTYP = 0). Use the higher 10 bits of the conversion result. Do not use the lower 2 bits.

<R> **2.** When using the converter with a resolution of 8 bits, select the 8-bit resolution mode (ADTYP = 1). Do not use the lower 4 bits of the ADCR register.
The higher 8 bits of the conversion result can be read by using the ADCRH register.

**Figure 11-9. Format of 12-bit A/D Conversion Result Register (ADCR)**

Address: FFF1EH, FFF1FH After reset: 0000H R

| Symbol | FFF1FH | | | | | | | | FFF1EH | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADCR | 0 | 0 | 0 | 0 | | | | | | | | | | | | |

**Cautions 1. When writing to the A/D converter mode register 0 (ADM0), analog input channel specification register (ADS), and A/D port configuration register (ADPC), the contents of the ADCR register may become undefined. Read the conversion result following conversion completion before writing to the ADM0, ADS, and ADPC registers. Using timing other than the above may cause an incorrect conversion result to be read.**

**2. If INTAD does not occur, the A/D conversion result is not stored in the ADCR register.**

### 11.3.6 8-bit A/D conversion result register (ADCRH)

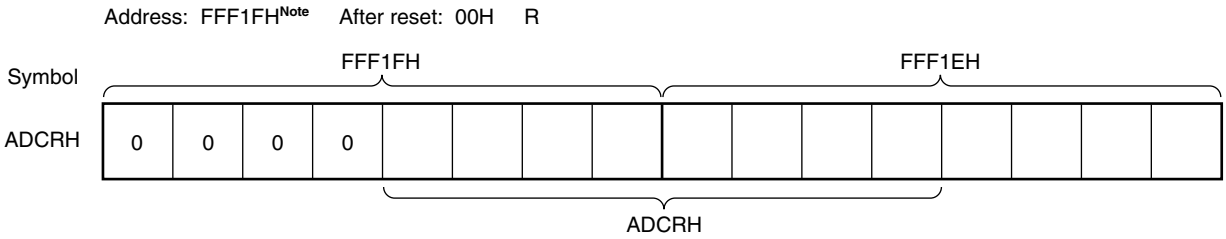
This register is an 8-bit register that indicate [11:4] bits of ADCR register. The higher 8 bits of 12-bit resolution are stored[Note].

The ADCRH register can be read by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Note** If the A/D conversion result is outside the range specified by using the A/D conversion comparison function (the value specified by the ADRCK bit of the ADM2 register and ADUL/ADLL registers; see **Figure 11-8**), the result is not stored.

**Figure 11-10. Format of 8-bit A/D Conversion Result Register (ADCRH)**




**Note** The ADCRH data (the lower 4 bits of FFF1FH + the higher 4 bits of FFF1EH) is to be read as a FFF1FH address.

**Cautions 1. When writing to the A/D converter mode register 0 (ADM0), analog input channel specification register (ADS), and A/D port configuration register (ADPC), the contents of the ADCRH register may become undefined. Read the conversion result following conversion completion before writing to the ADM0, ADS, and ADPC registers. Using timing other than the above may cause an incorrect conversion result to be read.**

**2. If INTAD does not occur, the A/D conversion result is not stored in the ADCRH register.**

### 11.3.7 Analog input channel specification register (ADS)

This register specifies the input channel of the analog voltage to be A/D converted.

The ADS register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 11-11. Format of Analog Input Channel Specification Register (ADS) (1/2)**

Address: FFF31H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADS | ADISS | 0 | 0 | ADS4 | ADS3 | ADS2 | ADS1 | ADS0 |

○ Select mode (ADMD = 0)

| ADISS | ADS4 | ADS3 | ADS2 | ADS1 | ADS0 | Analog input channel | Input source |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | ANI0 | P20/ANI0/$AV_{REFP}$ pin |
| 0 | 0 | 0 | 0 | 0 | 1 | ANI1 | P21/ANI1/$AV_{REFM}$ pin |
| 0 | 0 | 0 | 0 | 1 | 0 | ANI2 | P22/ANI2 pin |
| 0 | 0 | 0 | 0 | 1 | 1 | ANI3 | P23/ANI3 pin |
| 0 | 0 | 0 | 1 | 0 | 0 | ANI4 | P24/ANI4 pin |
| 0 | 0 | 0 | 1 | 0 | 1 | ANI5 | P25/ANI5 pin |
| 0 | 0 | 0 | 1 | 1 | 0 | ANI6 | P26/ANI6 pin |
| 0 | 0 | 0 | 1 | 1 | 1 | ANI7 | P27/ANI7 pin |
| 0 | 0 | 1 | 0 | 0 | 0 | ANI8 | P150/ANI8 pin |
| 0 | 0 | 1 | 0 | 0 | 1 | ANI9 | P151/ANI9 pin |
| 0 | 0 | 1 | 0 | 1 | 0 | ANI10 | P152/ANI10 pin |
| 0 | 0 | 1 | 0 | 1 | 1 | ANI11 | P153/ANI11 pin |
| 0 | 0 | 1 | 1 | 0 | 0 | ANI12 | P154/ANI12 pin |
| 0 | 0 | 1 | 1 | 0 | 1 | Setting prohibited | |
| 0 | 0 | 1 | 1 | 1 | 0 | Setting prohibited | |
| 0 | 0 | 1 | 1 | 1 | 1 | Setting prohibited | |
| 0 | 1 | 0 | 0 | 0 | 0 | ANI16 | P03/ANI16 pin |
| 0 | 1 | 0 | 0 | 0 | 1 | ANI17 | P02/ANI17 pin |
| 0 | 1 | 0 | 0 | 1 | 0 | ANI18 | P10/ANI18 pin |
| 0 | 1 | 0 | 0 | 1 | 1 | ANI19 | P120/ANI19 pin |
| 0 | 1 | 0 | 1 | 0 | 0 | ANI20 | P11/ANI20 pin |
| 0 | 1 | 0 | 1 | 0 | 1 | ANI21 | P12/ANI21 pin |
| 0 | 1 | 0 | 1 | 1 | 0 | ANI22 | P13/ANI22 pin |
| 0 | 1 | 0 | 1 | 1 | 1 | ANI23 | P14/ANI23 pin |
| 0 | 1 | 1 | 0 | 0 | 0 | ANI24 | P15/ANI24 pin |
| 0 | 1 | 1 | 0 | 0 | 1 | ANI25 | P51/ANI25 pin |
| 0 | 1 | 1 | 0 | 1 | 0 | ANI26 | P50/ANI26 pin |
| 0 | 1 | 1 | 0 | 1 | 1 | ANI27 | P30/ANI27 pin |
| 0 | 1 | 1 | 1 | 0 | 0 | ANI28 | P70/ANI28 pin |
| 0 | 1 | 1 | 1 | 0 | 1 | ANI29 | P31/ANI29 pin |
| 0 | 1 | 1 | 1 | 1 | 0 | ANI30 | P41/ANI30 pin |
| 0 | 1 | 1 | 1 | 1 | 1 | Setting prohibited | |
| 1 | 0 | 0 | 0 | 0 | 0 | – | Temperature sensor output voltage[Note] |
| 1 | 0 | 0 | 0 | 0 | 1 | – | Internal reference voltage (1.45 V)[Note] |
| Other than above | | | | | | Setting prohibited | |

**Note** This setting can be used only in HS (high-speed main) mode. For detail, see **Figure 24-3 Format of User Option Byte (000C2H/010C2H)**.

**Figure 11-11. Format of Analog Input Channel Specification Register (ADS) (2/2)**

Address: FFF31H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADS | ADISS | 0 | 0 | ADS4 | ADS3 | ADS2 | ADS1 | ADS0 |

○ Scan mode (ADMD = 1)

| ADISS | ADS4 | ADS3 | ADS2 | ADS1 | ADS0 | Analog input channel | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | Scan 0 | Scan 1 | Scan 2 | Scan 3 |
| 0 | 0 | 0 | 0 | 0 | 0 | ANI0 | ANI1 | ANI2 | ANI3 |
| 0 | 0 | 0 | 0 | 0 | 1 | ANI1 | ANI2 | ANI3 | ANI4 |
| 0 | 0 | 0 | 0 | 1 | 0 | ANI2 | ANI3 | ANI4 | ANI5 |
| 0 | 0 | 0 | 0 | 1 | 1 | ANI3 | ANI4 | ANI5 | ANI6 |
| 0 | 0 | 0 | 1 | 0 | 0 | ANI4 | ANI5 | ANI6 | ANI7 |
| 0 | 0 | 0 | 1 | 0 | 1 | ANI5 | ANI6 | ANI7 | ANI8 |
| 0 | 0 | 0 | 1 | 1 | 0 | ANI6 | ANI7 | ANI8 | ANI9 |
| 0 | 0 | 0 | 1 | 1 | 1 | ANI7 | ANI8 | ANI9 | ANI10 |
| 0 | 0 | 1 | 0 | 0 | 0 | ANI8 | ANI9 | ANI10 | ANI11 |
| 0 | 0 | 1 | 0 | 0 | 1 | ANI9 | ANI10 | ANI11 | ANI12 |
| 0 | 1 | 0 | 0 | 0 | 0 | ANI16 | ANI17 | ANI18 | ANI19 |
| 0 | 1 | 0 | 0 | 0 | 1 | ANI17 | ANI18 | ANI19 | ANI20 |
| 0 | 1 | 0 | 0 | 1 | 0 | ANI18 | ANI19 | ANI20 | ANI21 |
| 0 | 1 | 0 | 0 | 1 | 1 | ANI19 | ANI20 | ANI21 | ANI22 |
| 0 | 1 | 0 | 1 | 0 | 0 | ANI20 | ANI21 | ANI22 | ANI23 |
| 0 | 1 | 0 | 1 | 0 | 1 | ANI21 | ANI22 | ANI23 | ANI24 |
| 0 | 1 | 0 | 1 | 1 | 0 | ANI22 | ANI23 | ANI24 | ANI25 |
| 0 | 1 | 0 | 1 | 1 | 1 | ANI23 | ANI24 | ANI25 | ANI26 |
| 0 | 1 | 1 | 0 | 0 | 0 | ANI24 | ANI25 | ANI26 | ANI27 |
| 0 | 1 | 1 | 0 | 0 | 1 | ANI25 | ANI26 | ANI27 | ANI28 |
| 0 | 1 | 1 | 0 | 1 | 0 | ANI26 | ANI27 | ANI28 | ANI29 |
| 0 | 1 | 1 | 0 | 1 | 1 | ANI27 | ANI28 | ANI29 | ANI30 |
| Other than above | | | | | | Setting prohibited | | | |

**Cautions 1. Be sure to clear bits 5 and 6 to 0.**

**2 Set a channel to be set the analog input by ADPC and PMCx registers in the input mode by using port mode registers0 to 5, 7, 12, or 15 (PM0 to PM5, PM7, PM12, PM15).**

**3. Do not set the pin that is set by the A/D port configuration register (ADPC) as digital I/O by the ADS register.**

**4. Do not set the pin that is set by port mode control register 0, 1, 3 to 5, 7, or 12 (PMC0, PMC1, PMC3 to PMC5, PMC7, PMC12) as digital I/O by the ADS register.**

**5. Rewrite the value of the ADISS bit while conversion operation is stopped (ADCS = 0, ADCE = 0).**

**6. If using $AV_{REFP}$ as the + side reference voltage of the A/D converter, do not select ANI0 as an A/D conversion channel.**

**7. If using $AV_{REFM}$ as the – side reference voltage of the A/D converter, do not select ANI1 as an A/D conversion channel.**

**8. If ADISS is set to 1, the internal reference voltage (1.45 V) cannot be used for the + side reference voltage. Also, after setting the ADISS to 1, the result of the first conversion cannot be used. For details about the setting flow, see 11.7.4 Setup when temperature sensor output voltage/internal reference voltage is selected (example for software trigger mode and one-shot conversion mode).**

**9. Do not set the ADISS bit to 1 when shifting to STOP mode, or to HALT mode while the CPU is operating on the subsystem clock. When the ADISS bit is set to 1, the current value of the temperature sensor operating current ($I_{TMPS}$) and A/D converter reference voltage current ($I_{ADREF}$) indicated in 29.3.2 or 30.3.2 Supply current characteristics will be added.**

<R> **10. Ignore the conversion result if the corresponding ANI pin does not exist depending on the product (number of pins).**

### 11.3.8 Conversion result comparison upper limit setting register (ADUL)

This register is used to specify the setting for checking the upper limit of the A/D conversion results.

The A/D conversion results and ADUL register value are compared, and A/D conversion end interrupt request signal (INTAD) generation is controlled in the range specified for the ADRCK bit of A/D converter mode register 2 (ADM2) (shown in **Figure 11-8**).

The ADUL register can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets this register to FFH.

**Figure 11-12. Format of Conversion Result Comparison Upper Limit Setting Register (ADUL)**

Address: F0011H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADUL | ADUL7 | ADUL6 | ADUL5 | ADUL4 | ADUL3 | ADUL2 | ADUL1 | ADUL0 |

### 11.3.9 Conversion result comparison lower limit setting register (ADLL)

This register is used to specify the setting for checking the lower limit of the A/D conversion results.

The A/D conversion results and ADLL register value are compared, and A/D conversion end interrupt request signal (INTAD) generation is controlled in the range specified for the ADRCK bit of A/D converter mode register 2 (ADM2) (shown in **Figure 11-8**).

The ADLL register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 11-13. Format of Conversion Result Comparison Lower Limit Setting Register (ADLL)**

Address: F0012H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADLL | ADLL7 | ADLL6 | ADLL5 | ADLL4 | ADLL3 | ADLL2 | ADLL1 | ADLL0 |

**Cautions 1. When 12-bit resolution A/D conversion is selected, the higher eight bits of the 12-bit A/D conversion result register (ADCR) are compared with the ADUL register and ADLL register.**

**2. Only rewrite the value of the ADUL register and ADLL register while conversion operation is stopped (which is indicated by the ADCE bit of A/D converter mode register 0 (ADM0) being 0).**

**3. Make sure that ADUL > ADLL when setting these registers.**

### 11.3.10 A/D test register (ADTES)

<R> This register is used to select the + side reference voltage or – side reference voltage for the A/D converter, an analog input channel (ANIxx), the temperature sensor output voltage, or the internal reference voltage (1.45 V) as the target for A/D conversion. When using as the A/D test function, set as follows.

- For zero-scale measurement, select the – side reference voltage as the target for conversion.
- For full-scale measurement, select the + side reference voltage as the target for conversion.

The ADTES register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 11-14. Format of A/D Test Register (ADTES)**

Address: F0013H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADTES | 0 | 0 | 0 | 0 | 0 | 0 | ADTES1 | ADTES0 |

| ADTES1 | ADTES0 | A/D conversion target |
|---|---|---|
| 0 | 0 | ANIxx/temperature sensor output voltage[Note]/internal reference voltage (1.45 V)[Note] (This is specified using the analog input channel specification register (ADS).) |
| 1 | 0 | – side reference voltage (Setting at ADREFM bit of ADM2 register) |
| 1 | 1 | + side reference voltage (Setting at ADREFP1, ADREFP0 bits of ADM2 register) |
| Other than above | | Setting prohibited |

**Note** Temperature sensor output/internal reference voltage (1.45 V) can be used only in HS (high-speed main) mode. For detail, see **Figure 24-3 Format of User Option Byte (000C2H/010C2H)**.

<R> **Caution For details of the A/D test function, see CHAPTER 22 SAFETY FUNCTIONS.**

### <R> 11.3.11 Registers controlling port function of analog input pins

Set up the registers for controlling the functions of the ports shared with the analog input pins of the A/D converter (port mode registers (PMxx), port mode control registers (PMCxx), and A/D port configuration register (ADPC)).

For details, see as follows.

- **4.3.1 Port mode registers (PMxx)**
- **4.3.6 Port mode control registers (PMCxx)**
- **4.3.7 A/D port configuration register (ADPC)**.

When using the ANI0 to ANI12 pins for analog input of the A/D converter, set the port mode register (PMxx) bit corresponding to each port to 1 and select analog input through the A/D port configuration register (ADPC).

When using the ANI16 to ANI30 pins for analog input of the A/D converter, set the port mode register (PMxx) bit and port mode control register (PMCxx) bit corresponding to each port to 1.

## 11.4 A/D Converter Conversion Operations

The A/D converter conversion operations are described below.

<1> The voltage input to the selected analog input channel is sampled by the sample & hold circuit.

<2> When sampling has been done for a certain time, the sample & hold circuit is placed in the hold state and the sampled voltage is held until the A/D conversion operation has ended.

<3> Bit 11 of the successive approximation register (SAR) is set. The series resistor string voltage tap is set to (1/2) $AV_{REF}$ by the tap selector.

<4> The voltage difference between the series resistor string voltage tap and sampled voltage is compared by the voltage comparator. If the analog input is greater than (1/2) $AV_{REF}$, the MSB bit of the SAR register remains set to 1. If the analog input is smaller than (1/2) $AV_{REF}$, the MSB bit is reset to 0.

<5> Next, bit 10 of the SAR register is automatically set to 1, and the operation proceeds to the next comparison. The series resistor string voltage tap is selected according to the preset value of bit 11, as described below.

- Bit 11 = 1: (3/4) $AV_{REF}$
- Bit 11 = 0: (1/4) $AV_{REF}$

The voltage tap and sampled voltage are compared and bit 10 of the SAR register is manipulated as follows.

- Sampled voltage ≥ Voltage tap: Bit 10 = 1
- Sampled voltage < Voltage tap: Bit 10 = 0

<6> Comparison is continued in this way up to bit 0 of the SAR register.

<7> Upon completion of the comparison of 12 bits, an effective digital result value remains in the SAR register, and the result value is transferred to the A/D conversion result register (ADCR, ADCRH) and then latched[Note 1].
At the same time, the A/D conversion end interrupt request signal (INTAD) can also be generated[Note 1].

<8> Repeat steps <1> to <7>, until the ADCS bit is cleared to 0[Note 2].
To stop the A/D converter, clear the ADCS bit to 0.

**Notes 1.** If the A/D conversion result is outside the A/D conversion result range specified by the ADRCK bit and the ADUL and ADLL registers (see **Figure 11-8**), the A/D conversion result interrupt request signal is not generated and no A/D conversion results are stored in the ADCR and ADCRH registers.

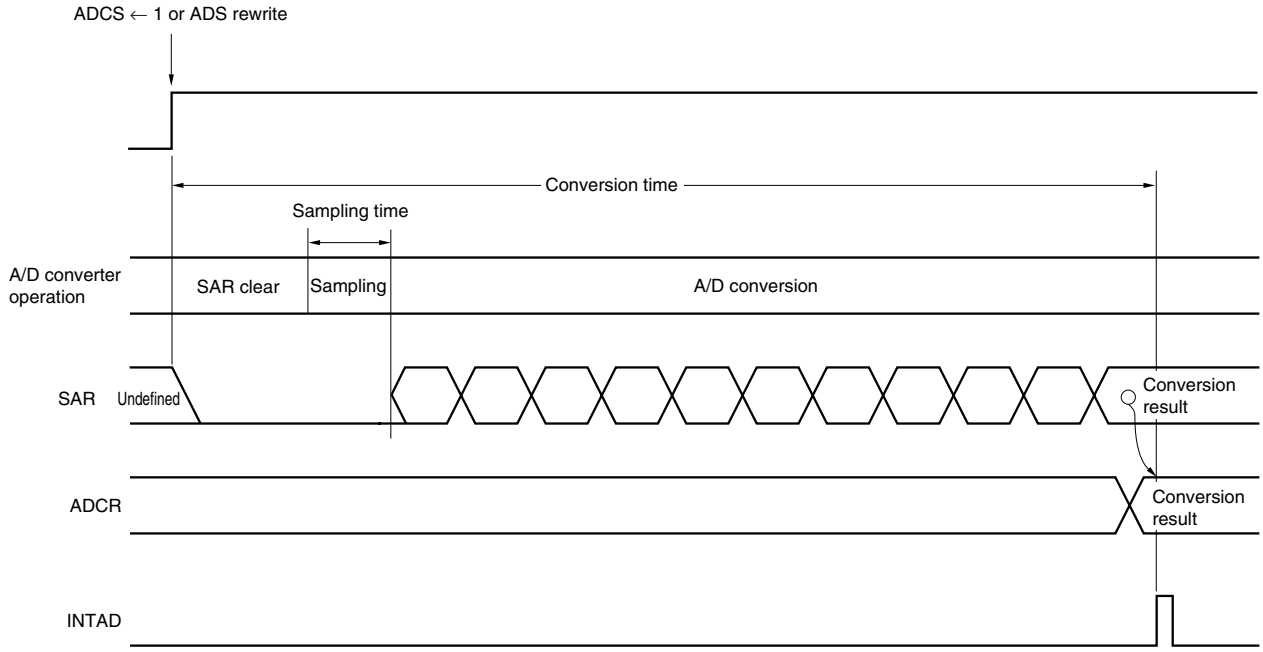
**2.** While in the sequential conversion mode, the ADCS flag is not automatically cleared to 0. This flag is not automatically cleared to 0 while in the one-shot conversion mode of the hardware trigger no-wait mode, either. Instead, 1 is retained.

**Remarks 1.** Two types of the A/D conversion result registers are available.

- ADCR register (16 bits): Store 12-bit A/D conversion value
- ADCRH register (8 bits): Store 8-bit A/D conversion value

**2.** $AV_{REF}$: The + side reference voltage of the A/D converter. This can be selected from $AV_{REFP}$, the internal reference voltage (1.45 V), and $AV_{DD}$.

**Figure 11-15. Conversion Operation of A/D Converter (Software Trigger Mode)**




<R> In one-shot conversion mode, the ADCS bit is automatically cleared to 0 after completion of A/D conversion.

In sequential conversion mode, A/D conversion operations proceed continuously until the software clears bit 7 (ADCS) of the A/D converter mode register 0 (ADM0) to 0.

Rewriting and overwriting to the analog input channel specification register (ADS) during A/D conversion interrupts the current conversion after which A/D conversion of the analog input specified by the ADS register proceeds. Data from the A/D conversion that was in progress are discarded.

Reset signal generation clears the A/D conversion result register (ADCR, ADCRH) to 0000H or 00H.

## 11.5 Input Voltage and Conversion Results

The relationship between the analog input voltage input to the analog input pins (ANI0 to ANI12, ANI16 to ANI30) and the theoretical A/D conversion result (stored in the 12-bit A/D conversion result register (ADCR)) is shown by the following expression.

$$ADCR = INT\left(\frac{V_{AIN}}{AV_{REF}} \times 4096 + 0.5\right)$$

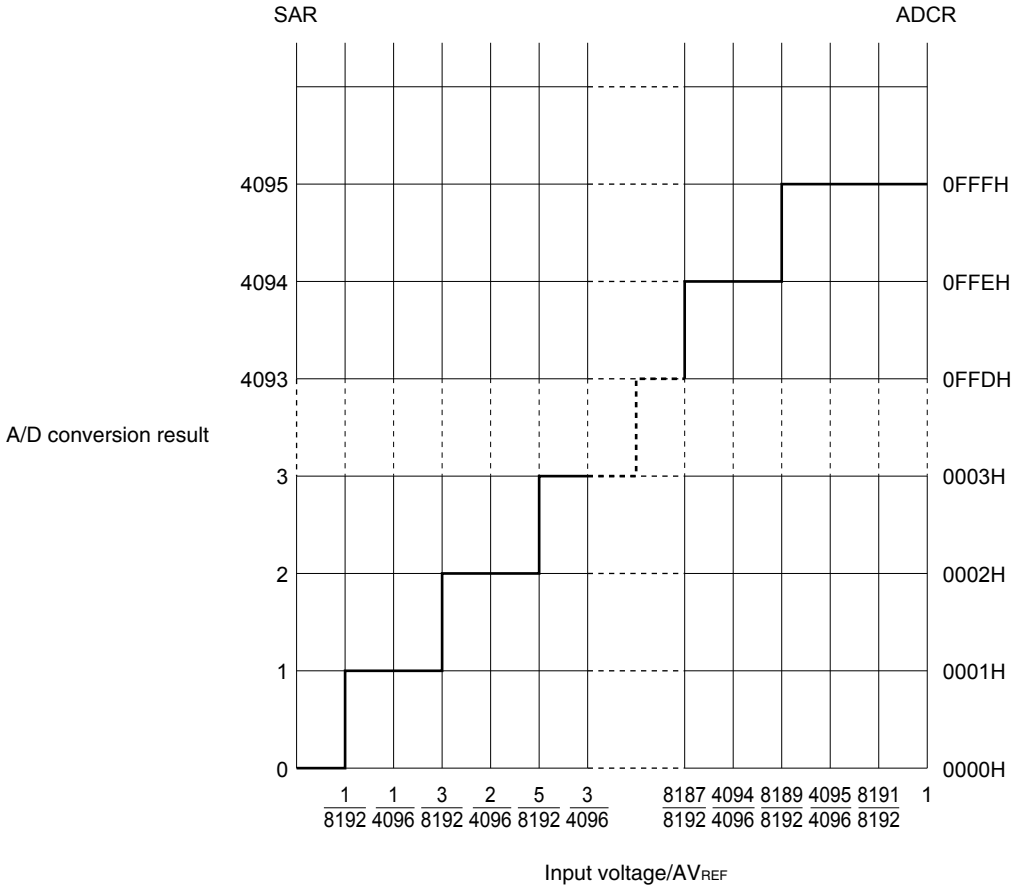
or

$$(ADCR - 0.5) \times \frac{AV_{REF}}{4096} \leq V_{AIN} < (ADCR + 0.5) \times \frac{AV_{REF}}{4096}$$

where, INT( ): Function which returns integer part of value in parentheses
$V_{AIN}$: Analog input voltage
$AV_{REF}$: $AV_{REF}$ pin voltage
ADCR: A/D conversion result register (ADCR) value

Figure 11-16 shows the relationship between the analog input voltage and the A/D conversion result.

**Figure 11-16. Relationship Between Analog Input Voltage and A/D Conversion Result**




**Remark** $AV_{REF}$: The + side reference voltage of the A/D converter. This can be selected from $AV_{REFP}$, the internal reference voltage (1.45 V), and $AV_{DD}$.

## 11.6 A/D Converter Operation Modes

The operation of each A/D converter mode is described below. In addition, the procedure for specifying each mode is described in **11.7 A/D Converter Setup Flowchart**.

### 11.6.1 Software trigger mode (select mode, sequential conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.

<2> After the software counts up to the stabilization wait time[Note], the ADCS bit of the ADM0 register is set to 1 to perform the A/D conversion of the analog input specified by the analog input channel specification register (ADS).

<3> When A/D conversion ends, the conversion result is stored in the A/D conversion result register (ADCR, ADCRH), and the A/D conversion end interrupt request signal (INTAD) is generated. After A/D conversion ends, the next A/D conversion immediately starts.

<4> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.

<5> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input respecified by the ADS register. The partially converted data is discarded.

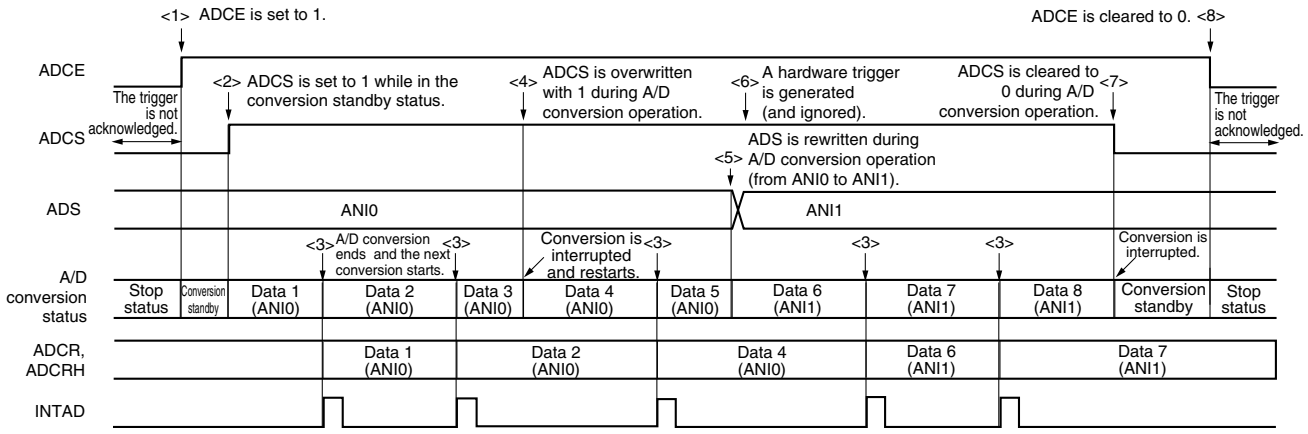
<6> Even if a hardware trigger is input during conversion operation, A/D conversion does not start.

<7> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status.

<8> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCE = 0, specifying 1 for ADCS is ignored and A/D conversion does not start.

**Note** If a high-accuracy channel is selected as the analog input channel: Stabilization wait time = 0.5 $\mu$s
If a standard channel is selected as the analog input channel: Stabilization wait time = 2 $\mu$s

**Figure 11-17. Example of Software Trigger Mode (Select Mode, Sequential Conversion Mode) Operation Timing**



### 11.6.2 Software trigger mode (select mode, one-shot conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.

<2> After the software counts up to the stabilization wait time[Note], the ADCS bit of the ADM0 register is set to 1 to perform the A/D conversion of the analog input specified by the analog input channel specification register (ADS).

<3> When A/D conversion ends, the conversion result is stored in the A/D conversion result register (ADCR, ADCRH), and the A/D conversion end interrupt request signal (INTAD) is generated.

<4> After A/D conversion ends, the ADCS bit is automatically cleared to 0, and the system enters the A/D conversion standby status.

<5> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.

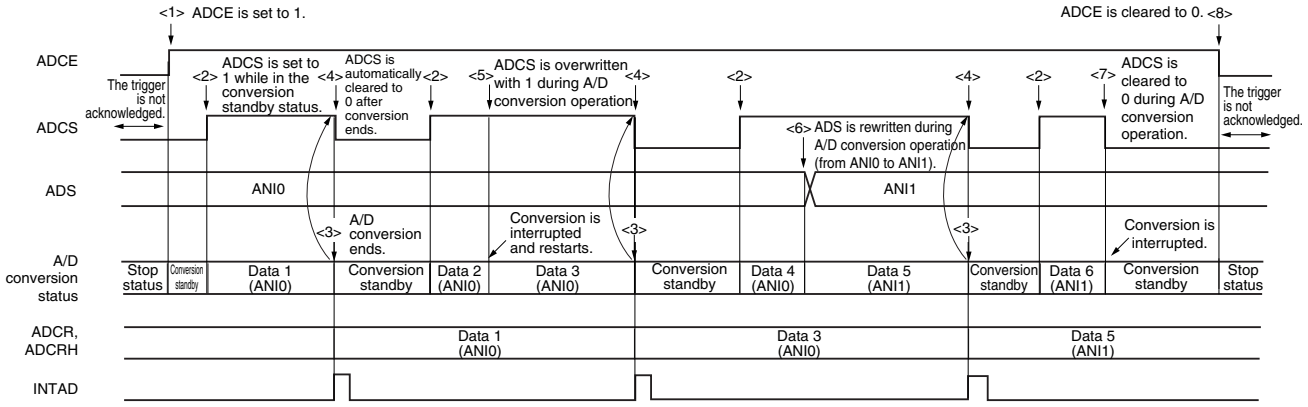
<6> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input respecified by the ADS register. The partially converted data is discarded.

<7> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status.

<8> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCE = 0, specifying 1 for ADCS is ignored and A/D conversion does not start. In addition, A/D conversion does not start even if a hardware trigger is input while in the A/D conversion standby status.

**Note** If a high-accuracy channel is selected as the analog input channel: Stabilization wait time = 0.5 μs
If a standard channel is selected as the analog input channel: Stabilization wait time = 2 μs

**Figure 11-18. Example of Software Trigger Mode (Select Mode, One-Shot Conversion Mode) Operation Timing**



### 11.6.3 Software trigger mode (scan mode, sequential conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.

<2> After the software counts up to the stabilization wait time[Note], the ADCS bit of the ADM0 register is set to 1 to perform A/D conversion on the four analog input channels specified by scan 0 to scan 3, which are specified by the analog input channel specification register (ADS). A/D conversion is performed on the analog input channels in order, starting with that specified by scan 0.

<3> A/D conversion is sequentially performed on the four analog input channels, the conversion results are stored in the A/D conversion result register (ADCR, ADCRH) each time conversion ends, and the A/D conversion end interrupt request signal (INTAD) is generated. After A/D conversion of the four channels ends, the A/D conversion of the channel following the specified channel automatically starts (until all four channels are finished).

<4> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.

<5> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register. The partially converted data is discarded.

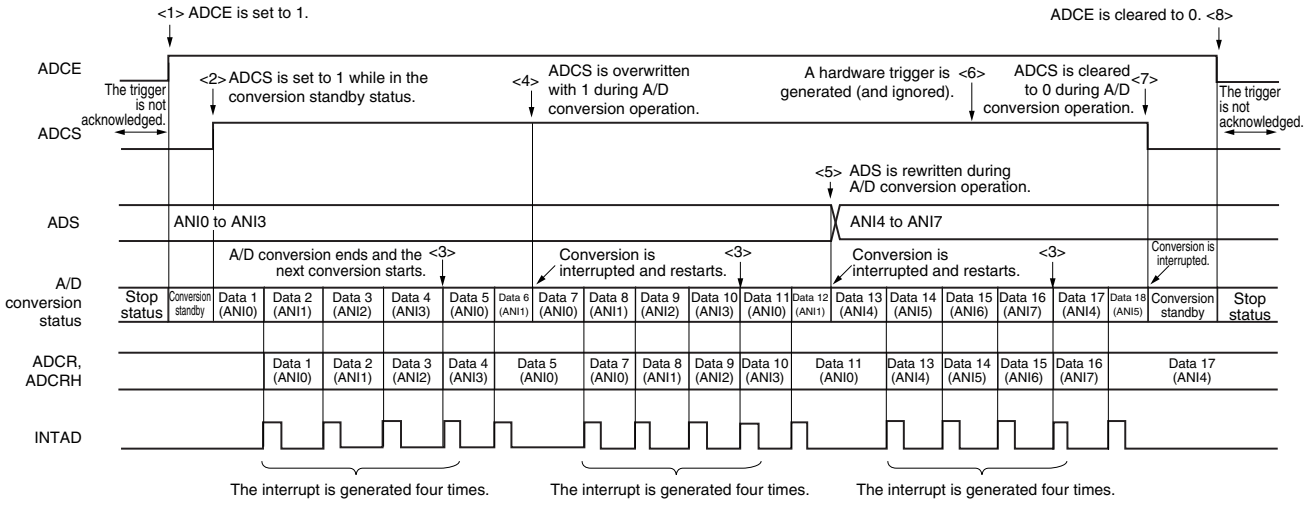
<6> Even if a hardware trigger is input during conversion operation, A/D conversion does not start.

<7> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status.

<8> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCE = 0, specifying 1 for ADCS is ignored and A/D conversion does not start.

**Note** If a high-accuracy channel is selected as the analog input channel: Stabilization wait time = 0.5 $\mu$s
If a standard channel is selected as the analog input channel: Stabilization wait time = 2 $\mu$s

**Figure 11-19. Example of Software Trigger Mode (Scan Mode, Sequential Conversion Mode) Operation Timing**



### 11.6.4 Software trigger mode (scan mode, one-shot conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.

<2> After the software counts up to the stabilization wait time[Note], the ADCS bit of the ADM0 register is set to 1 to perform A/D conversion on the four analog input channels specified by scan 0 to scan 3, which are specified by the analog input channel specification register (ADS). A/D conversion is performed on the analog input channels in order, starting with that specified by scan 0.

<3> A/D conversion is sequentially performed on the four analog input channels, the conversion results are stored in the A/D conversion result register (ADCR, ADCRH) each time conversion ends, and the A/D conversion end interrupt request signal (INTAD) is generated.

<4> After A/D conversion of the four channels ends, the ADCS bit is automatically cleared to 0, and the system enters the A/D conversion standby status.

<5> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.

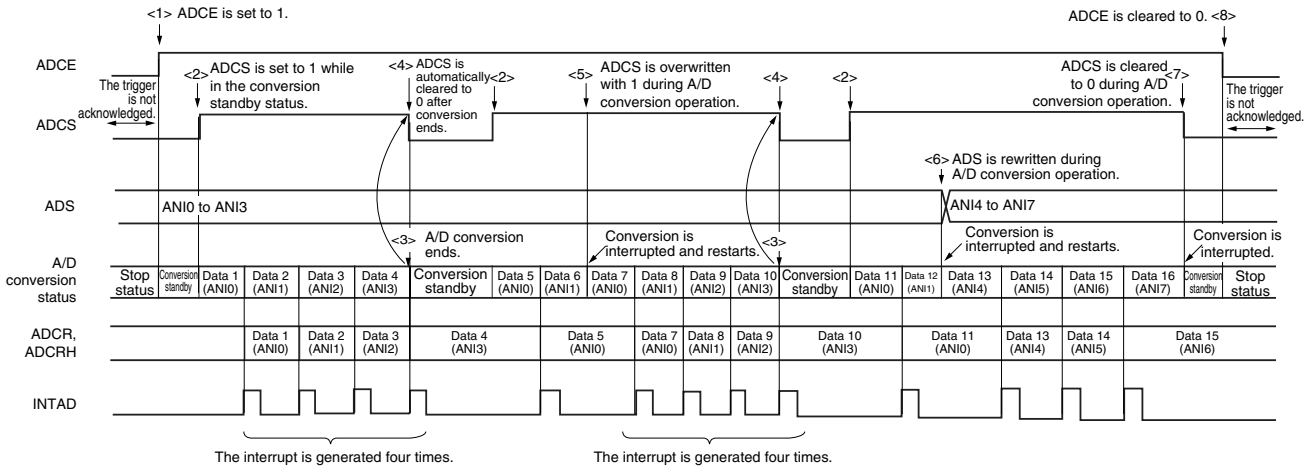
<6> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register. The partially converted data is discarded.

<7> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status.

<8> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCE = 0, specifying 1 for ADCS is ignored and A/D conversion does not start. In addition, A/D conversion does not start even if a hardware trigger is input while in the A/D conversion standby status.

**Note** If a high-accuracy channel is selected as the analog input channel: Stabilization wait time = 0.5 μs
If a standard channel is selected as the analog input channel: Stabilization wait time = 2 μs

**Figure 11-20. Example of Software Trigger Mode (Scan Mode, One-Shot Conversion Mode) Operation Timing**



### 11.6.5 Hardware trigger no-wait mode (select mode, sequential conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.

<2> After the software counts up to the stabilization wait time[Note], the ADCS bit of the ADM0 register is set to 1 to place the system in the hardware trigger standby status (and conversion does not start at this stage). Note that, while in this status, A/D conversion does not start even if ADCS is set to 1.

<3> If a hardware trigger is input while ADCS = 1, A/D conversion is performed on the analog input specified by the analog input channel specification register (ADS).

<4> When A/D conversion ends, the conversion result is stored in the A/D conversion result register (ADCR, ADCRH), and the A/D conversion end interrupt request signal (INTAD) is generated. After A/D conversion ends, the next A/D conversion immediately starts.

<5> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.

<6> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input respecified by the ADS register. The partially converted data is discarded.

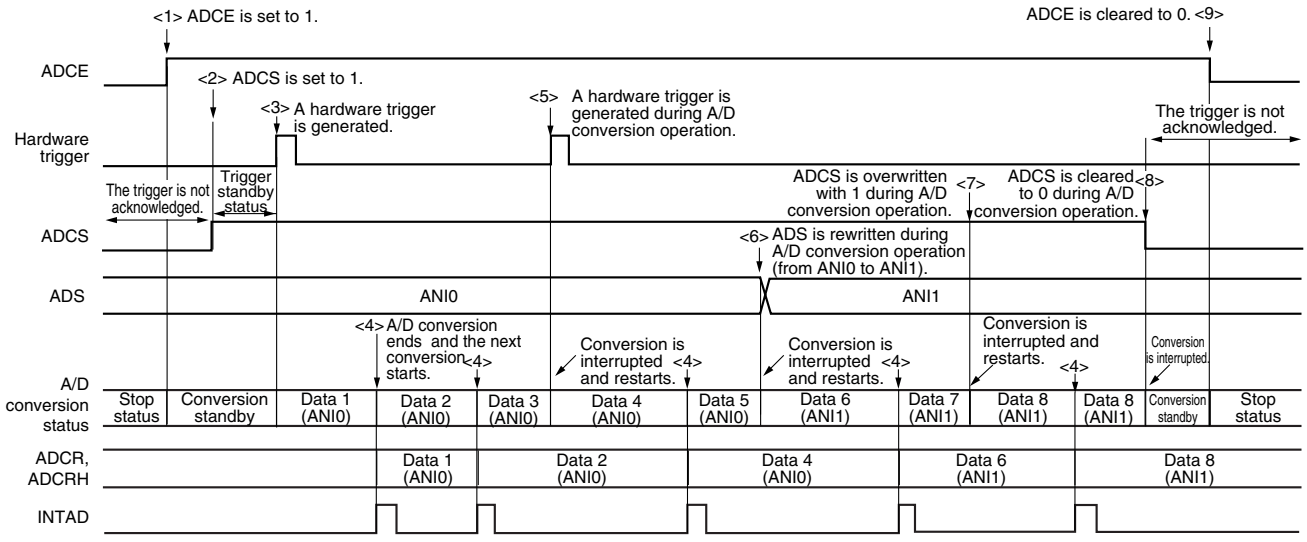
<7> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.

<8> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status. However, the A/D converter does not stop in this status.

<9> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCS = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

**Note** If a high-accuracy channel is selected as the analog input channel: Stabilization wait time = 0.5 μs
If a standard channel is selected as the analog input channel: Stabilization wait time = 2 μs

**Figure 11-21. Example of Hardware Trigger No-Wait Mode (Select Mode, Sequential Conversion Mode) Operation Timing**



### 11.6.6 Hardware trigger no-wait mode (select mode, one-shot conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.

<2> After the software counts up to the stabilization wait time[Note], the ADCS bit of the ADM0 register is set to 1 to place the system in the hardware trigger standby status (and conversion does not start at this stage). Note that, while in this status, A/D conversion does not start even if ADCS is set to 1.

<3> If a hardware trigger is input while ADCS = 1, A/D conversion is performed on the analog input specified by the analog input channel specification register (ADS).

<4> When A/D conversion ends, the conversion result is stored in the A/D conversion result register (ADCR, ADCRH), and the A/D conversion end interrupt request signal (INTAD) is generated.

<5> After A/D conversion ends, the ADCS bit remains set to 1, and the system enters the A/D conversion standby status.

<6> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.

<7> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input respecified by the ADS register. The partially converted data is discarded.

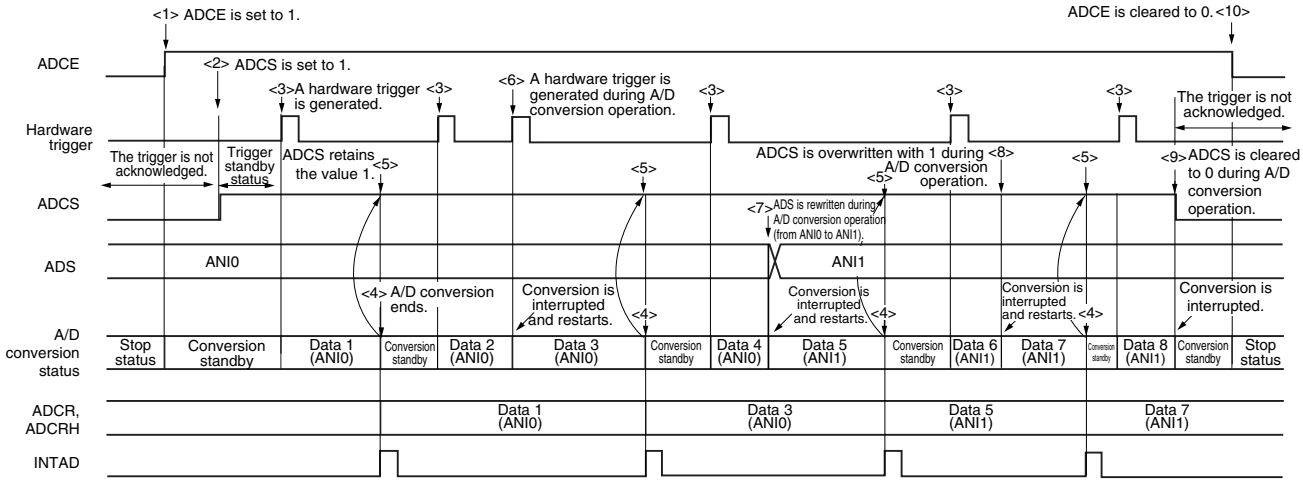
<8> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.

<9> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status. However, the A/D converter does not stop in this status.

<10> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCS = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

**Note** If a high-accuracy channel is selected as the analog input channel: Stabilization wait time = 0.5 $\mu$s
If a standard channel is selected as the analog input channel: Stabilization wait time = 2 $\mu$s

**Figure 11-22. Example of Hardware Trigger No-Wait Mode (Select Mode, One-Shot Conversion Mode) Operation Timing**



### 11.6.7 Hardware trigger no-wait mode (scan mode, sequential conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.

<2> After the software counts up to the stabilization wait time[Note], the ADCS bit of the ADM0 register is set to 1 to place the system in the hardware trigger standby status (and conversion does not start at this stage). Note that, while in this status, A/D conversion does not start even if ADCS is set to 1.

<3> If a hardware trigger is input while ADCS = 1, A/D conversion is performed on the four analog input channels specified by scan 0 to scan 3, which are specified by the analog input channel specification register (ADS). A/D conversion is performed on the analog input channels in order, starting with that specified by scan 0.

<4> A/D conversion is sequentially performed on the four analog input channels, the conversion results are stored in the A/D conversion result register (ADCR, ADCRH) each time conversion ends, and the A/D conversion end interrupt request signal (INTAD) is generated. After A/D conversion of the four channels ends, the A/D conversion of the channel following the specified channel automatically starts.

<5> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.

<6> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register. The partially converted data is discarded.

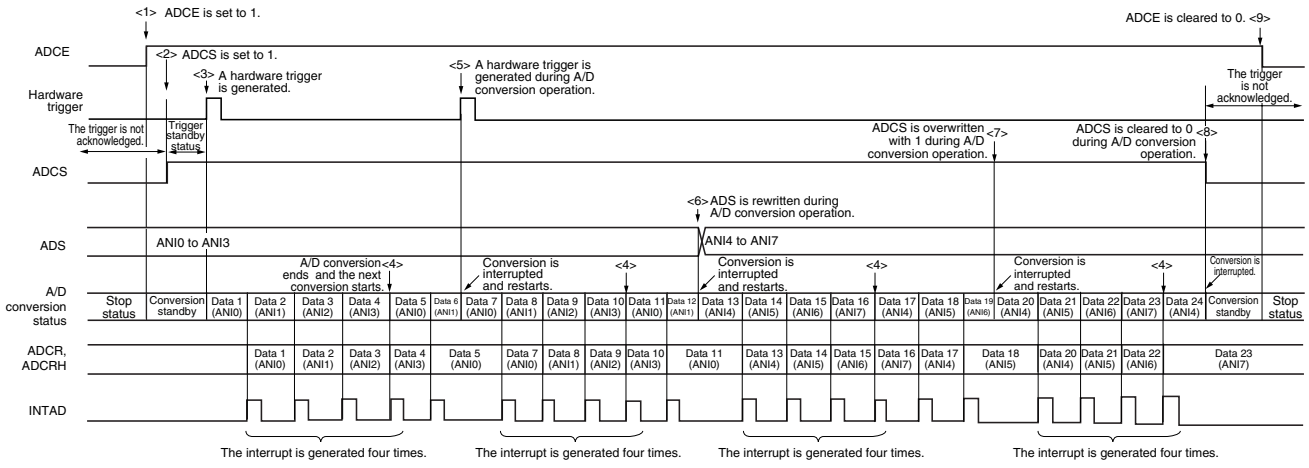
<7> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.

<8> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status. However, the A/D converter does not stop in this status.

<9> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCE = 0, specifying 1 for ADCS is ignored and A/D conversion does not start.

**Note** If a high-accuracy channel is selected as the analog input channel: Stabilization wait time = 0.5 $\mu$s
If a standard channel is selected as the analog input channel: Stabilization wait time = 2 $\mu$s

**Figure 11-23. Example of Hardware Trigger No-Wait Mode (Scan Mode, Sequential Conversion Mode) Operation Timing**



### 11.6.8 Hardware trigger no-wait mode (scan mode, one-shot conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.

<2> After the software counts up to the stabilization wait time[Note], the ADCS bit of the ADM0 register is set to 1 to place the system in the hardware trigger standby status (and conversion does not start at this stage). Note that, while in this status, A/D conversion does not start even if ADCS is set to 1.

<3> If a hardware trigger is input while ADCS = 1, A/D conversion is performed on the four analog input channels specified by scan 0 to scan 3, which are specified by the analog input channel specification register (ADS). A/D conversion is performed on the analog input channels in order, starting with that specified by scan 0.

<4> A/D conversion is sequentially performed on the four analog input channels, the conversion results are stored in the A/D conversion result register (ADCR, ADCRH) each time conversion ends, and the A/D conversion end interrupt request signal (INTAD) is generated.

<5> After A/D conversion of the four channels ends, the ADCS bit remains set to 1, and the system enters the A/D conversion standby status.

<6> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.

<7> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register. The partially converted data is discarded.

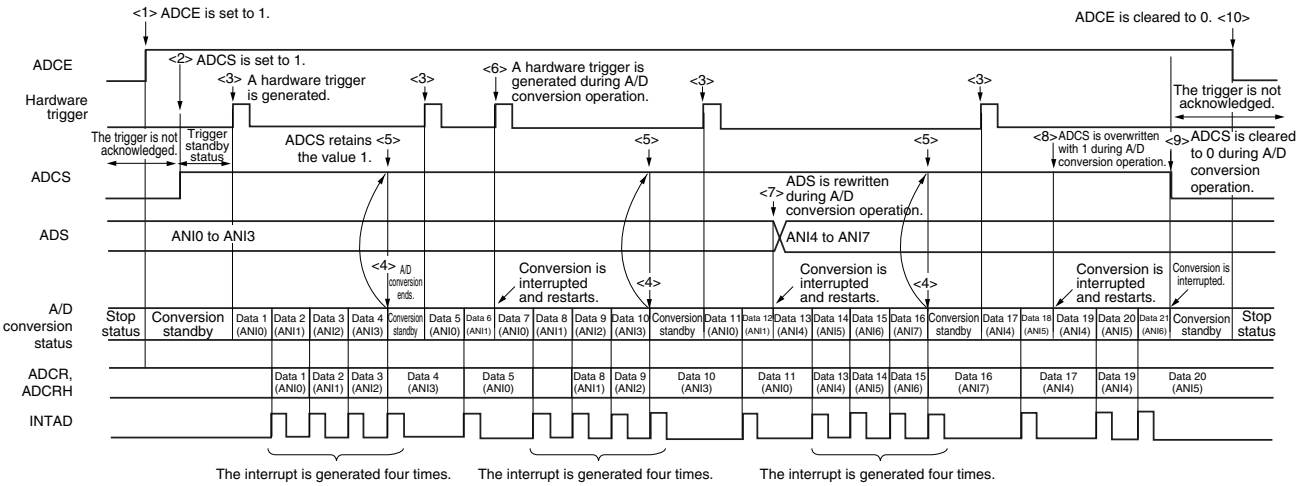
<8> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register. The partially converted data is discarded.

<9> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status. However, the A/D converter does not stop in this status.

<10> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCS = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

**Note** If a high-accuracy channel is selected as the analog input channel: Stabilization wait time = 0.5 μs
If a standard channel is selected as the analog input channel: Stabilization wait time = 2 μs

**Figure 11-24. Example of Hardware Trigger No-Wait Mode (Scan Mode, One-Shot Conversion Mode) Operation Timing**



### 11.6.9 Hardware trigger wait mode (select mode, sequential conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the hardware trigger standby status.

<2> If a hardware trigger is input while in the hardware trigger standby status, A/D conversion is performed on the analog input specified by the analog input channel specification register (ADS). The ADCS bit of the ADM0 register is automatically set to 1 according to the hardware trigger input.

<3> When A/D conversion ends, the conversion result is stored in the A/D conversion result register (ADCR, ADCRH), and the A/D conversion end interrupt request signal (INTAD) is generated. After A/D conversion ends, the next A/D conversion immediately starts. (At this time, no hardware trigger is necessary.)

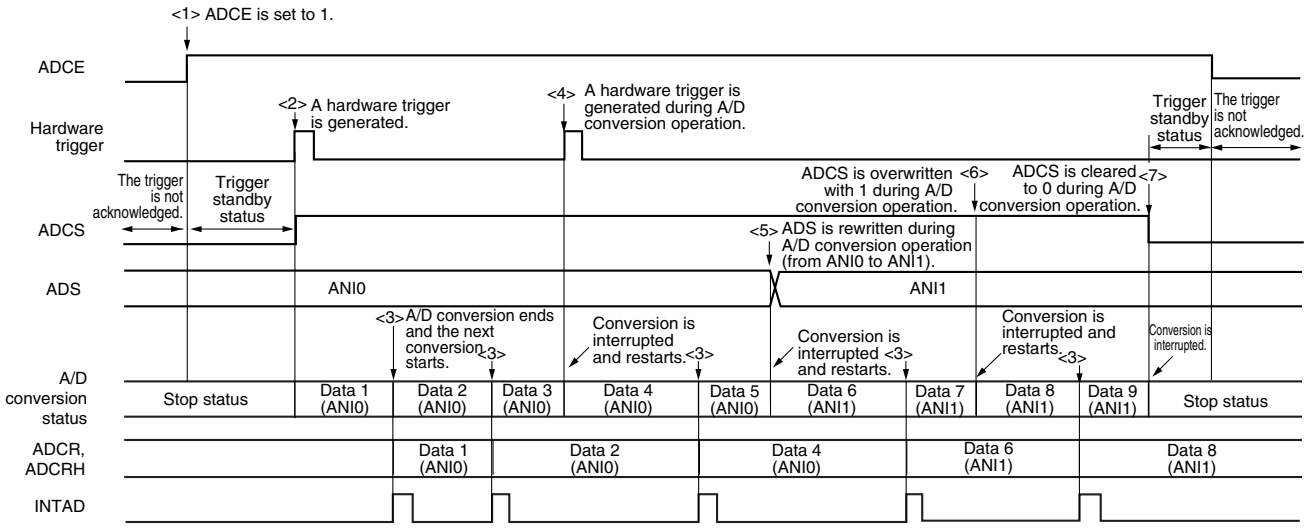
<4> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.

<5> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input respecified by the ADS register. The partially converted data is discarded.

<6> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.

<7> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, the system enters the hardware trigger standby status, and the A/D converter enters the stop status. When ADCE = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

**Figure 11-25. Example of Hardware Trigger Wait Mode (Select Mode, Sequential Conversion Mode) Operation Timing**



### 11.6.10 Hardware trigger wait mode (select mode, one-shot conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the hardware trigger standby status.

<2> If a hardware trigger is input while in the hardware trigger standby status, A/D conversion is performed on the analog input specified by the analog input channel specification register (ADS). The ADCS bit of the ADM0 register is automatically set to 1 according to the hardware trigger input.

<3> When A/D conversion ends, the conversion result is stored in the A/D conversion result register (ADCR, ADCRH), and the A/D conversion end interrupt request signal (INTAD) is generated.

<4> After A/D conversion ends, the ADCS bit is automatically cleared to 0, and the A/D converter enters the stop status.

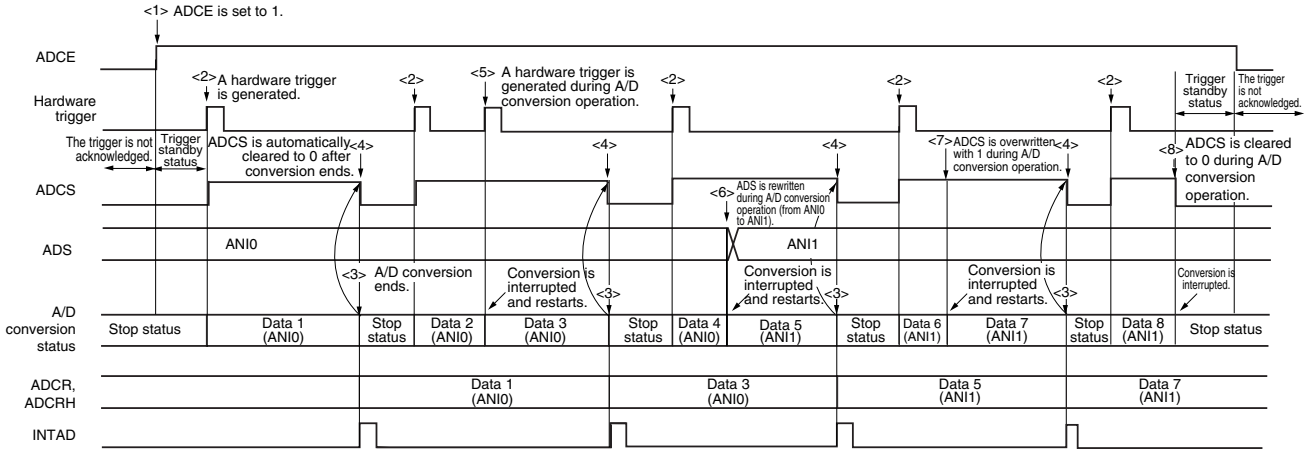
<5> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.

<6> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input respecified by the ADS register. The partially converted data is discarded.

<7> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is initialized.

<8> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, the system enters the hardware trigger standby status, and the A/D converter enters the stop status. When ADCE = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

**Figure 11-26. Example of Hardware Trigger Wait Mode (Select Mode, One-Shot Conversion Mode) Operation Timing**



### 11.6.11 Hardware trigger wait mode (scan mode, sequential conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.

<2> If a hardware trigger is input while in the hardware trigger standby status, A/D conversion is performed on the four analog input channels specified by scan 0 to scan 3, which are specified by the analog input channel specification register (ADS). The ADCS bit of the ADM0 register is automatically set to 1 according to the hardware trigger input. A/D conversion is performed on the analog input channels in order, starting with that specified by scan 0.

<3> A/D conversion is sequentially performed on the four analog input channels, the conversion results are stored in the A/D conversion result register (ADCR, ADCRH) each time conversion ends, and the A/D conversion end interrupt request signal (INTAD) is generated. After A/D conversion of the four channels ends, the A/D conversion of the channel following the specified channel automatically starts.

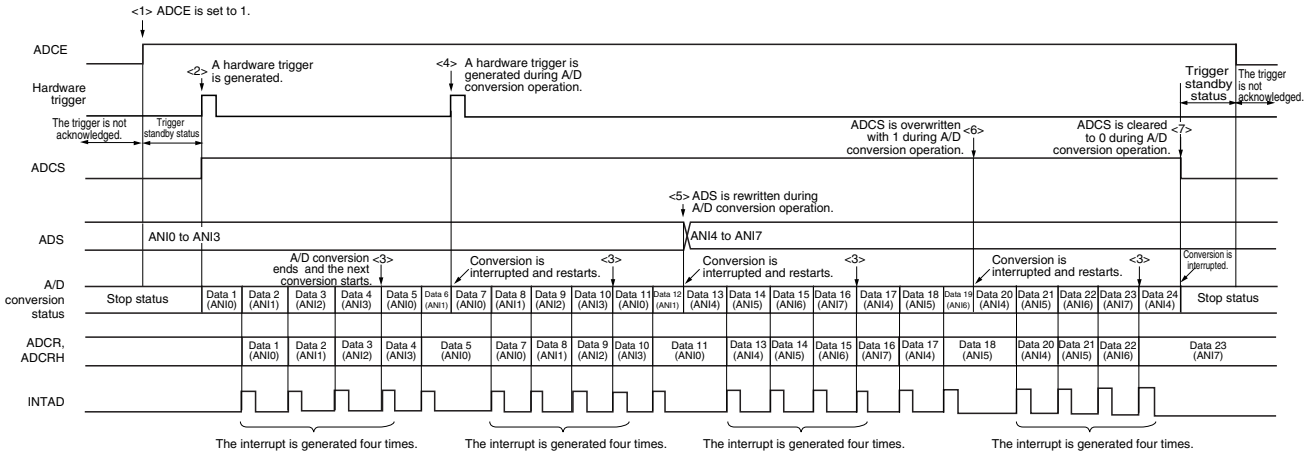
<4> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.

<5> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register. The partially converted data is discarded.

<6> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.

<7> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, the system enters the hardware trigger standby status, and the A/D converter enters the stop status. When ADCE = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

**Figure 11-27. Example of Hardware Trigger Wait Mode (Scan Mode, Sequential Conversion Mode) Operation Timing**



### 11.6.12 Hardware trigger wait mode (scan mode, one-shot conversion mode)

<1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.

<2> If a hardware trigger is input while in the hardware trigger standby status, A/D conversion is performed on the four analog input channels specified by scan 0 to scan 3, which are specified by the analog input channel specification register (ADS). The ADCS bit of the ADM0 register is automatically set to 1 according to the hardware trigger input. A/D conversion is performed on the analog input channels in order, starting with that specified by scan 0.

<3> A/D conversion is sequentially performed on the four analog input channels, the conversion results are stored in the A/D conversion result register (ADCR, ADCRH) each time conversion ends, and the A/D conversion end interrupt request signal (INTAD) is generated.

<4> After A/D conversion ends, the ADCS bit is automatically cleared to 0, and the A/D converter enters the stop status.

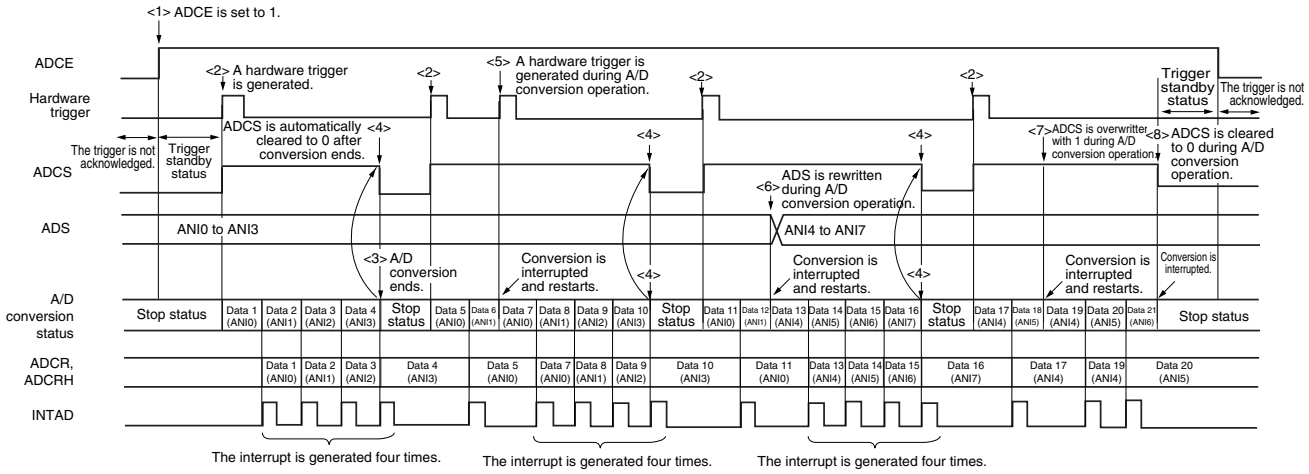
<5> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.

<6> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register. The partially converted data is discarded.

<7> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.

<8> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, the system enters the hardware trigger standby status, and the A/D converter enters the stop status. When ADCE = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

**Figure 11-28. Example of Hardware Trigger Wait Mode (Scan Mode, One-Shot Conversion Mode) Operation Timing**



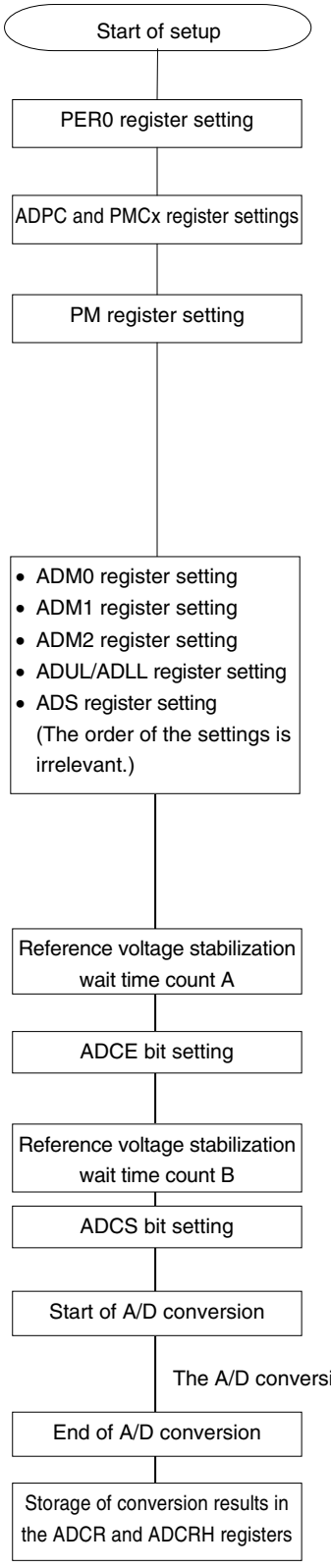
## 11.7 A/D Converter Setup Flowchart

The A/D converter setup flowchart in each operation mode is described below.

### 11.7.1 Setting up software trigger mode

<R>

**Figure 11-29. Setting up Software Trigger Mode**




The ADCEN bit of the PER0 register is set (1), and supplying the clock starts.

The ports are set to analog input.
ANI0 to ANI12 pins: Set using the ADPC register
ANI16 to ANI30 pins: Set using the PMCx register

The ports are set to the input mode.

- ADM0 register
  FR2 to FR0, LV1, and LV0 bits: These are used to specify the A/D conversion time.
  ADMD bit: Select mode/scan mode
- ADM1 register
  ADTMD1 and ADTMD0 bits: These are used to specify the software trigger mode.
  ADSCM bit: Sequential conversion mode/one-shot conversion mode
- ADM2 register
  ADREFP1, ADREFP0, and ADREFM bits: These are used to select the reference voltage.
  ADRCK bit: This is used to select the range for the A/D conversion result comparison value generated by the interrupt signal from AREA1, AREA3, and AREA2.
  ADTYP bit: 8-bit/12-bit resolution
- ADUL/ADLL register
  These are used to specify the upper limit and lower limit A/D conversion result comparison values.
- ADS register
  ADS4 to ADS0 bits: These are used to select the analog input channels.

The reference voltage stabilization wait time count A is required when the value of the ADREFP1 and ADREFP0 bits is changed.
If change the ADREFP1 and ADREFP0 = 1, 0: A = 10 $\mu$s
If change the ADREFP1 and ADREFP0 = 0, 0 or 0, 1: A = 1 $\mu$s

The ADCE bit of the ADM0 register is set (1), and the system enters the A/D conversion standby status.

If a high-accuracy channel is selected as the analog input channel: B = 0.5 $\mu$s
If a standard channel is selected as the analog input channel: B = 2 $\mu$s

After completion of the reference voltage stabilization wait time count B, the ADCS bit of the ADM0 register is set (1), and A/D conversion starts.

The A/D conversion operations are performed.

The A/D conversion end interrupt request signal (INTAD) is generated.[Note]

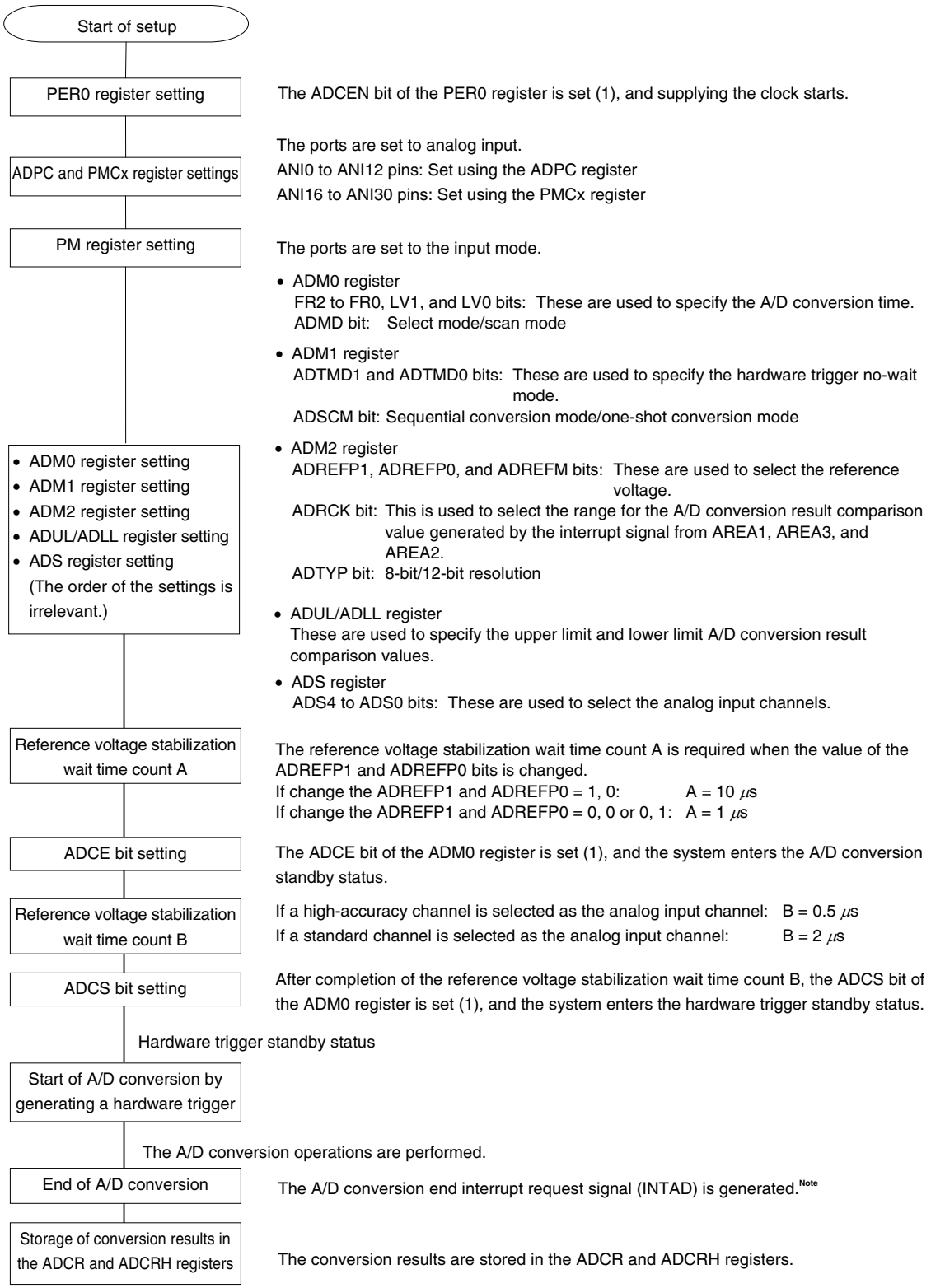
The conversion results are stored in the ADCR and ADCRH registers.

**Note** Depending on the settings of the ADRCK bit and ADUL/ADLL register, there is a possibility of no A/D conversion end interrupt request signal (INTAD) being generated. In this case, the results are not stored in the ADCR, ADCRH registers.

### 11.7.2 Setting up hardware trigger no-wait mode

<R>

**Figure 11-30. Setting up Hardware Trigger No-Wait Mode**

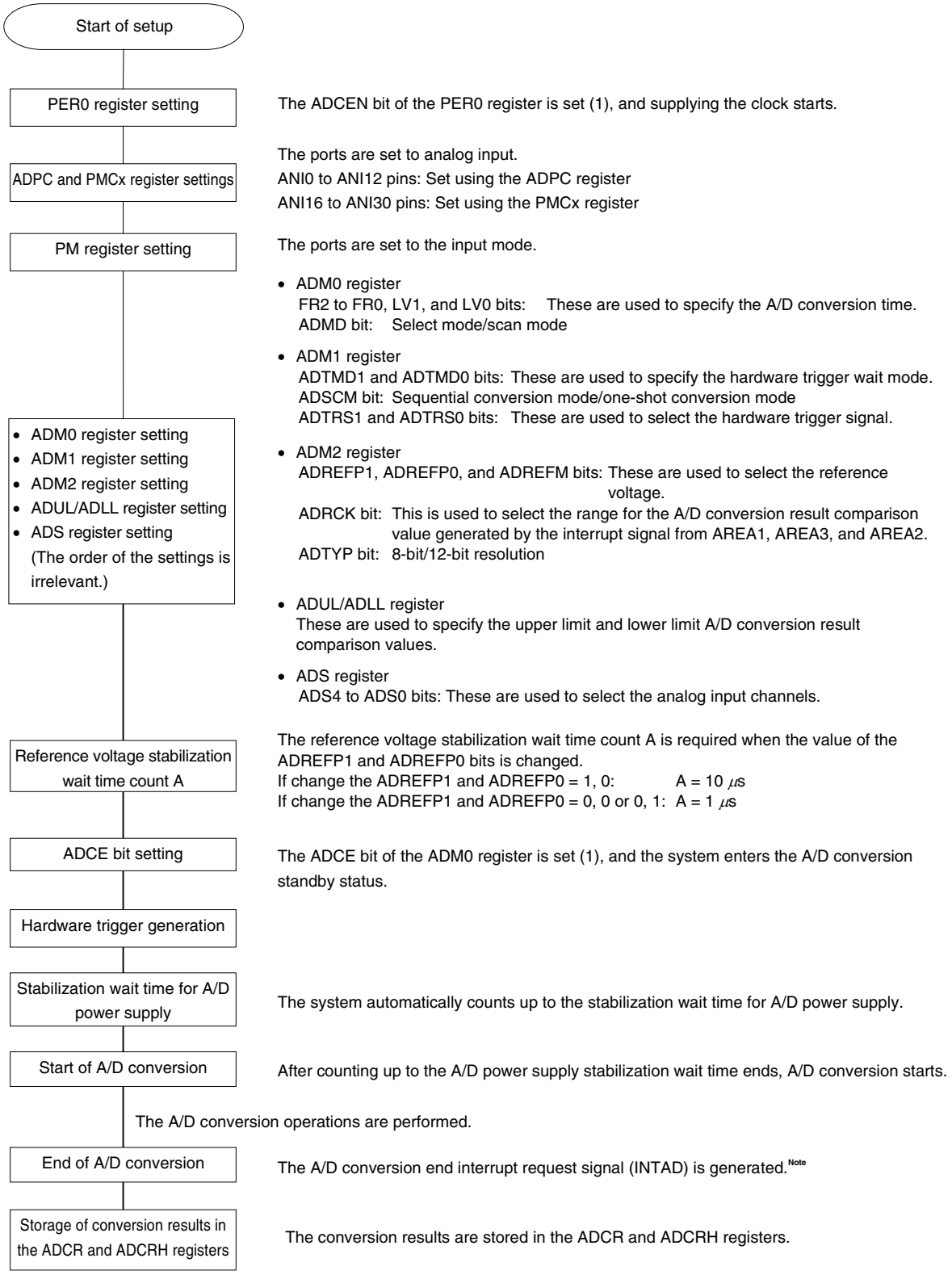



**Note** Depending on the settings of the ADRCK bit and ADUL/ADLL register, there is a possibility of no A/D conversion end interrupt request signal (INTAD) being generated. In this case, the results are not stored in the ADCR, ADCRH registers.

### 11.7.3 Setting up hardware trigger wait mode

<R>

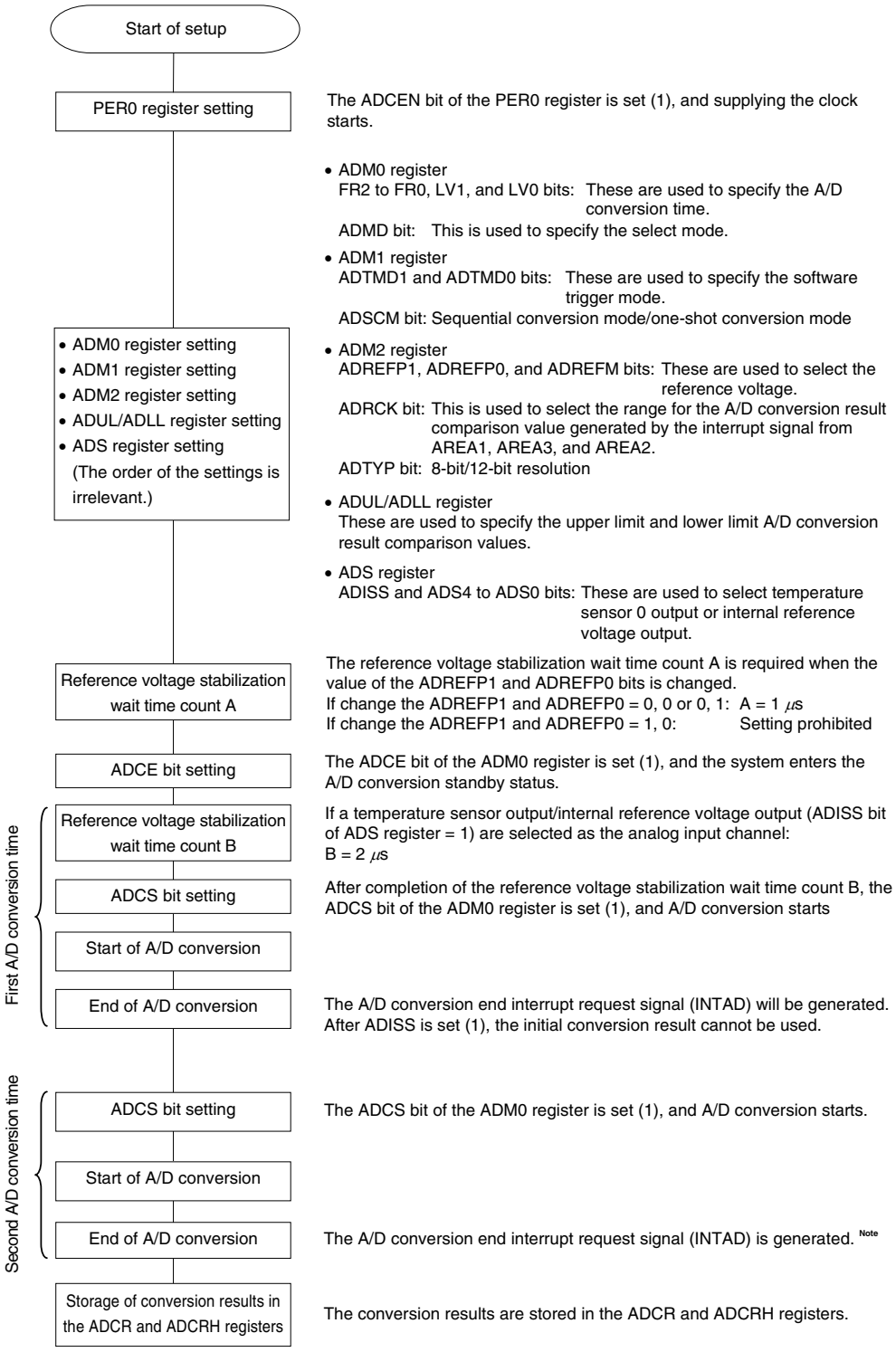
**Figure 11-31. Setting up Hardware Trigger Wait Mode**




**Note** Depending on the settings of the ADRCK bit and ADUL/ADLL register, there is a possibility of no A/D conversion end interrupt request signal (INTAD) being generated. In this case, the results are not stored in the ADCR, ADCRH registers.

### 11.7.4 Setup when temperature sensor output voltage/internal reference voltage is selected (example for software trigger mode and one-shot conversion mode)

<R> **Figure 11-32. Setup When Temperature Sensor Output Voltage/Internal Reference Voltage Is Selected**

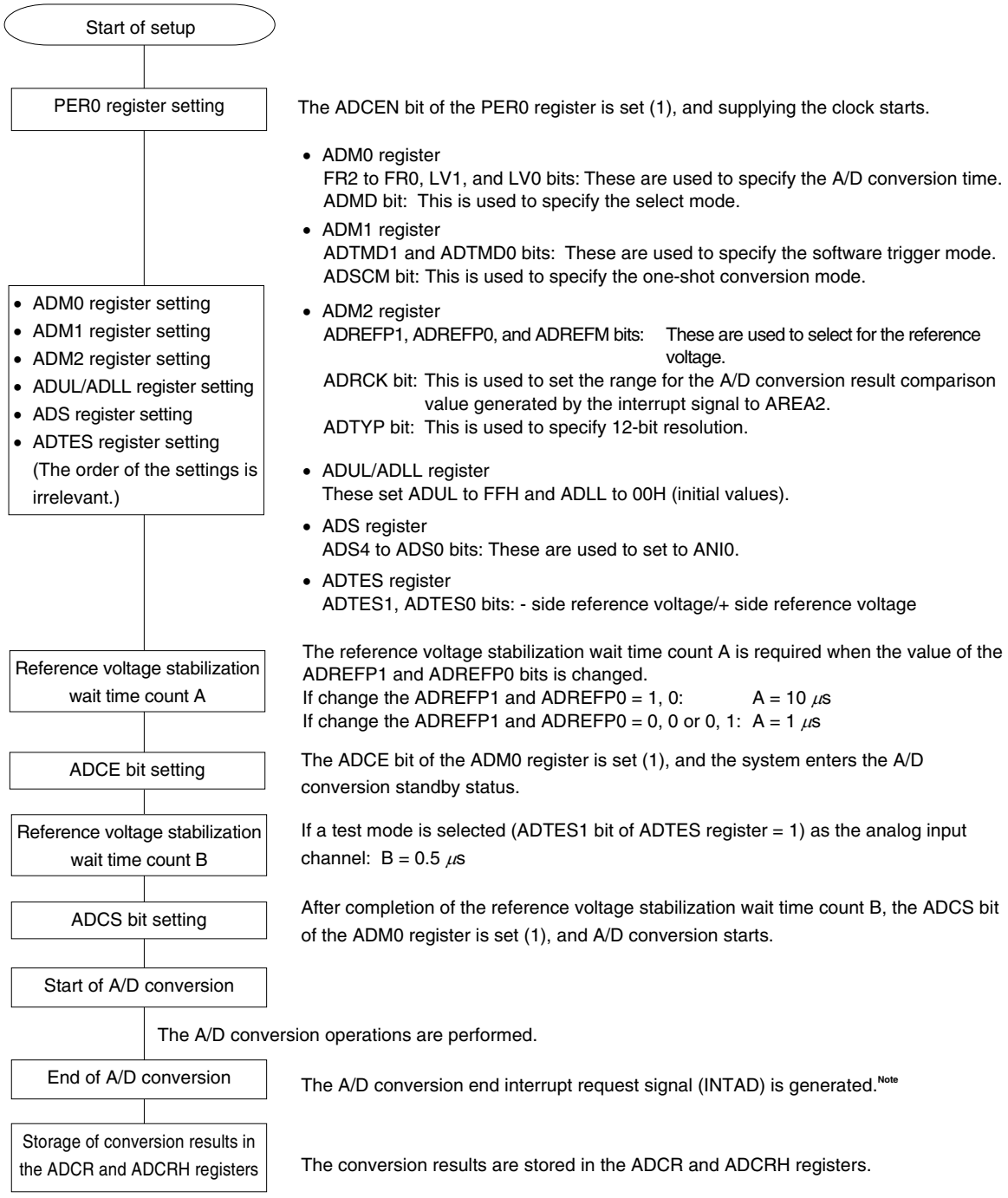



**Note** Depending on the settings of the ADRCK bit and ADUL/ADLL register, there is a possibility of no A/D conversion end interrupt request signal (INTAD) being generated. In this case, the results are not stored in the ADCR, ADCRH registers.

**Caution This setting can be used only in HS (high-speed main) mode. For detail, see Figure 24-3 Format of User Option Byte (000C2H/010C2H).**

### 11.7.5 Setting up test mode

<R> **Figure 11-33. Setting up Test Mode**




**Note** Depending on the settings of the ADRCK bit and ADUL/ADLL register, there is a possibility of no A/D conversion end interrupt request signal (INTAD) being generated. In this case, the results are not stored in the ADCR, ADCRH registers.

<R> **Caution For the procedure for testing the A/D converter, see 22.3.8 A/D test function.**

## 11.8 SNOOZE Mode Function

In the SNOOZE mode, A/D conversion is triggered by inputting a hardware trigger in the STOP mode. Normally, A/D conversion is stopped while in the STOP mode, but, by using the SNOOZE mode function, A/D conversion can be performed without operating the CPU. This is effective for reducing the operating current.

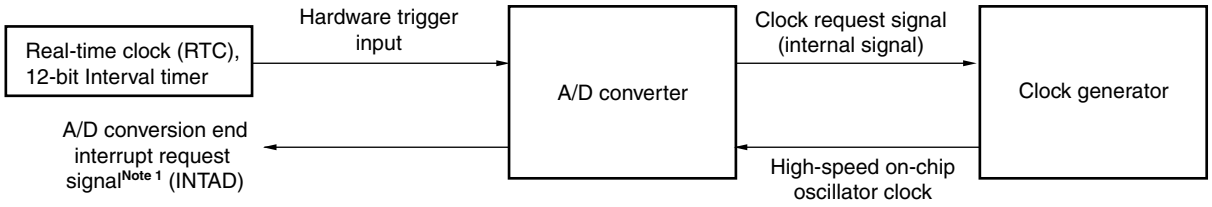
If the A/D conversion result range is specified using the ADUL and ADLL registers, A/D conversion results can be judged at a certain interval of time in SNOOZE mode. Using this function enables power supply voltage monitoring and input key judgment based on A/D inputs.

In the SNOOZE mode, only the following two conversion modes can be used:

- Hardware trigger wait mode (select mode, one-shot conversion mode)
- Hardware trigger wait mode (scan mode, one-shot conversion mode)

**Caution The SNOOZE mode can only be specified when the high-speed on-chip oscillator clock is selected for $f_{CLK}$.**

**Figure 11-34. Block Diagram When Using SNOOZE Mode Function**




When using the SNOOZE mode function, the initial setting of each register is specified before switching to the STOP mode. (For details about these settings, see **11.7.3 Setting up hardware trigger wait mode** Note 2.) Just before move to STOP mode, bit 2 (AWC) of A/D converter mode register 2 (ADM2) is set to 1. After the initial settings are specified, bit 0 (ADCE) of A/D converter mode register 0 (ADM0) is set to 1.

If a hardware trigger is input after switching to the STOP mode, the high-speed on-chip oscillator clock is supplied to the A/D converter. After supplying this clock, the system automatically counts up to the A/D power supply stabilization wait time, and then A/D conversion starts.

The SNOOZE mode operation after A/D conversion ends differs depending on whether an interrupt signal is generated Note 1.

**Notes 1.** Depending on the setting of the A/D conversion result comparison function (ADRCK bit, ADUL/ADLL register), there is a possibility of no interrupt signal being generated.

**2.** Be sure to set the ADM1 register to E2H or E3H.

**Remark** The hardware trigger is INTRTC or INTIT.
Specify the hardware trigger by using the A/D Converter Mode Register 1 (ADM1).

**(1) If an interrupt is generated after A/D conversion ends**

If the A/D conversion result value is inside the range of values specified by the A/D conversion result comparison function (which is set up by using the ADRCK bit and ADUL/ADLL register), the A/D conversion end interrupt request signal (INTAD) is generated.

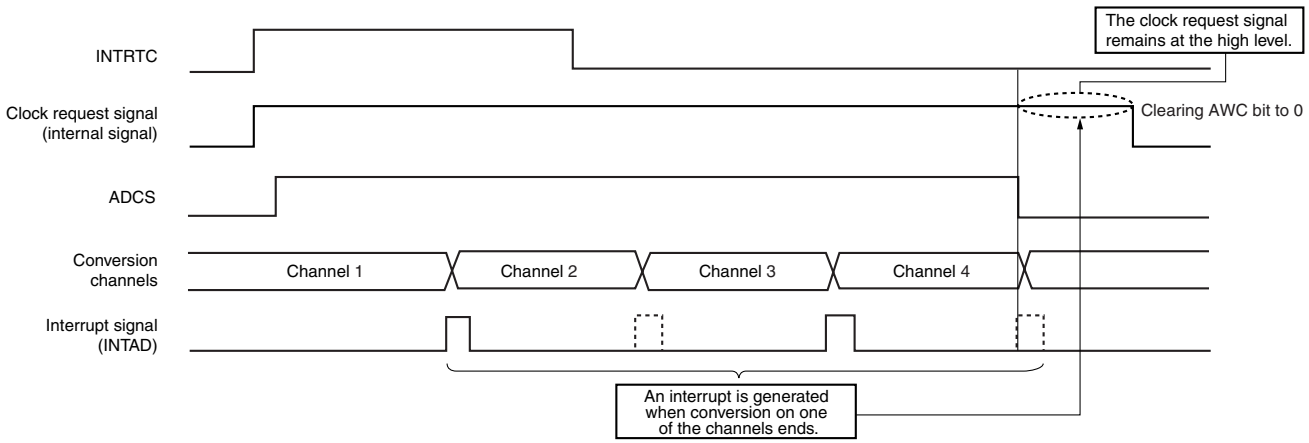
- While in the select mode

  When A/D conversion ends and an A/D conversion end interrupt request signal (INTAD) is generated, the A/D converter returns to normal operation mode from SNOOZE mode. At this time, be sure to clear bit 2 (AWC = 0: SNOOZE mode release) of the A/D converter mode register 2 (ADM2). If the AWC bit is left set to 1, A/D conversion will not start normally in the subsequent SNOOZE or normal operation mode.

- While in the scan mode

  If even one A/D conversion end interrupt request signal (INTAD) is generated during A/D conversion of the four channels, the clock request signal remains at the high level, and the A/D converter switches from the SNOOZE mode to the normal operation mode. At this time, be sure to clear bit 2 (AWC = 0: SNOOZE mode release) of A/D converter mode register 2 (ADM2) to 0. If the AWC bit is left set to 1, A/D conversion will not start normally in the subsequent SNOOZE or normal operation mode.

**Figure 11-35. Operation Example When Interrupt Is Generated After A/D Conversion Ends (While in Scan Mode)**



**(2) If no interrupt is generated after A/D conversion ends**

If the A/D conversion result value is outside the range of values specified by the A/D conversion result comparison function (which is set up by using the ADRCK bit and ADUL/ADLL register), the A/D conversion end interrupt request signal (INTAD) is not generated.

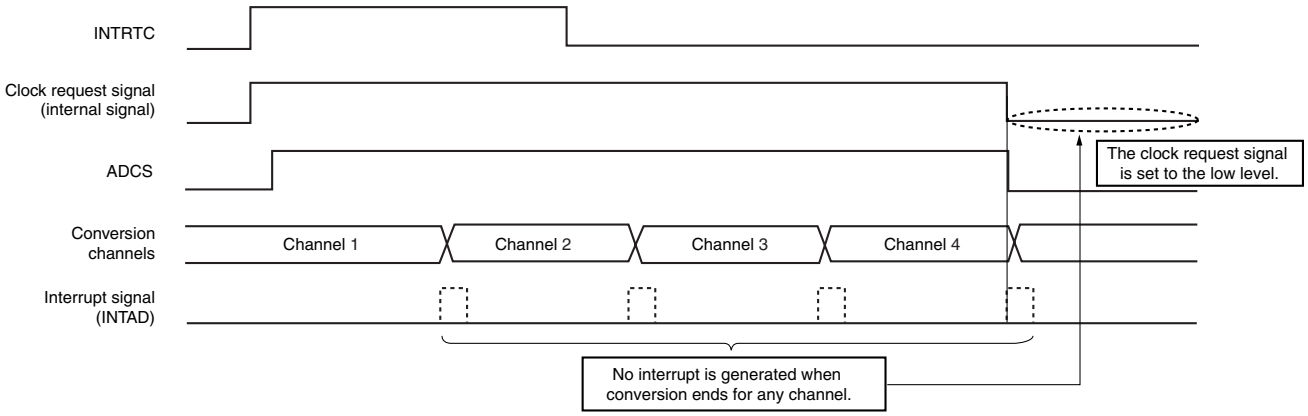
- While in the select mode

  If the A/D conversion end interrupt request signal (INTAD) is not generated after A/D conversion ends, the clock request signal (an internal signal) is automatically set to the low level, and supplying the high-speed on-chip oscillator clock stops. If a hardware trigger is input later, A/D conversion work is again performed in the SNOOZE mode.

- While in the scan mode

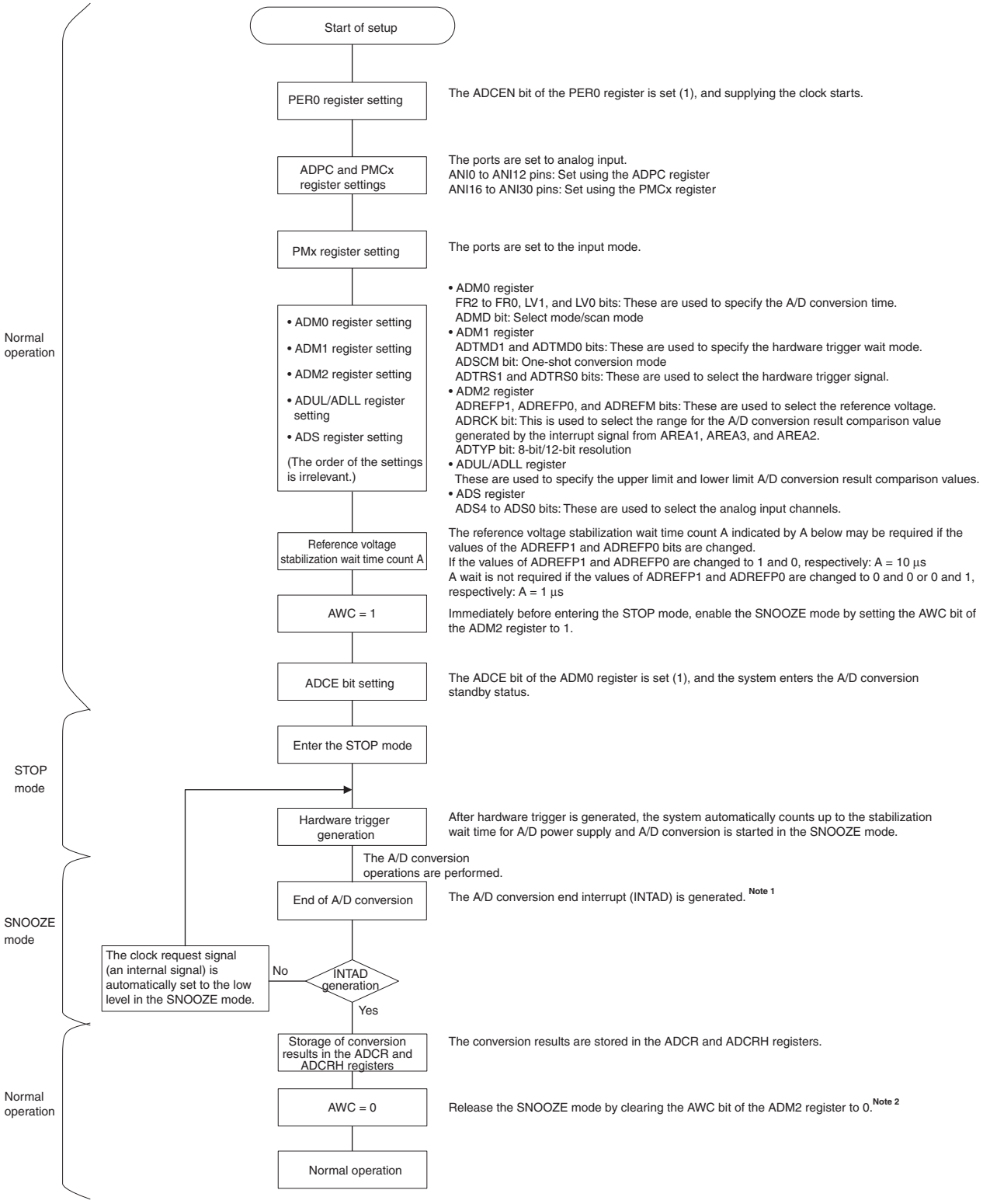
  If the A/D conversion end interrupt request signal (INTAD) is not generated even once during A/D conversion of the four channels, the clock request signal (an internal signal) is automatically set to the low level after A/D conversion of the four channels ends, and supplying the high-speed on-chip oscillator clock stops. If a hardware trigger is input later, A/D conversion work is again performed in the SNOOZE mode.

**Figure 11-36. Operation Example When No Interrupt Is Generated After A/D Conversion Ends (While in Scan Mode)**



<R>

**Figure 11-37. Flowchart for Setting up SNOOZE Mode**




**Notes 1.** If the A/D conversion end interrupt request signal (INTAD) is not generated by setting ADRCK bit and ADUL/ADLL register, the result is not stored in the ADCR and ADCRH registers. The system enters the STOP mode again. If a hardware trigger is input later, A/D conversion operation is again performed in the SNOOZE mode.

**2.** If the AWC bit is left set to 1, A/D conversion will not start normally in spite of the subsequent SNOOZE or normal operation mode. Be sure to clear the AWC bit to 0.

## 11.9 How to Read A/D Converter Characteristics Table

Here, special terms unique to the A/D converter are explained.

### (1) Resolution

This is the minimum analog input voltage that can be identified. That is, the percentage of the analog input voltage per bit of digital output is called 1LSB (Least Significant Bit). The percentage of 1LSB with respect to the full scale is expressed by %FSR (Full Scale Range).

1LSB is as follows when the resolution is 12 bits.

$1LSB = 1/2^{12} = 1/4096$
$\approx 0.024\%FSR$

Accuracy has no relation to resolution, but is determined by overall error.

### (2) Overall error

This shows the maximum error value between the actual measured value and the theoretical value.
Zero-scale error, full-scale error, integral linearity error, and differential linearity errors that are combinations of these express the overall error.
Note that the quantization error is not included in the overall error in the characteristics table.

### (3) Quantization error

When analog values are converted to digital values, a ±1/2LSB error naturally occurs. In an A/D converter, an analog input voltage in a range of ±1/2LSB is converted to the same digital code, so a quantization error cannot be avoided.
Note that the quantization error is not included in the overall error, zero-scale error, full-scale error, integral linearity error, and differential linearity error in the characteristics table.

**Figure 11-38. Overall Error**

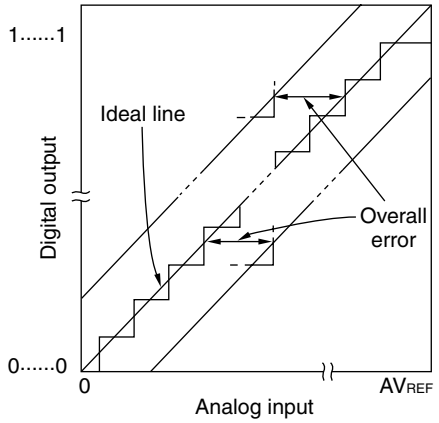
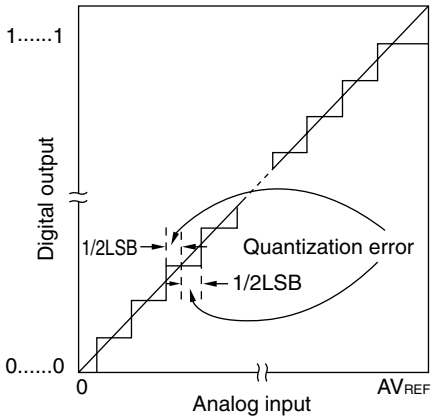



**Figure 11-39. Quantization Error**




### (4) Zero-scale error

This shows the difference between the actual measurement value of the analog input voltage and the theoretical value (1/2LSB) when the digital output changes from 0.....000 to 0.....001.
If the actual measurement value is greater than the theoretical value, it shows the difference between the actual measurement value of the analog input voltage and the theoretical value (3/2LSB) when the digital output changes from 0……001 to 0……010.

**(5) Full-scale error**

This shows the difference between the actual measurement value of the analog input voltage and the theoretical value (Full-scale – 3/2LSB) when the digital output changes from 1.....110 to 1.....111.

**(6) Integral linearity error**

This shows the degree to which the conversion characteristics deviate from the ideal linear relationship. It expresses the maximum value of the difference between the actual measurement value and the ideal straight line when the zero-scale error and full-scale error are 0.

**(7) Differential linearity error**

While the ideal width of code output is 1LSB, this indicates the difference between the actual measurement value and the ideal value.

**Figure 11-40. Zero-Scale Error**

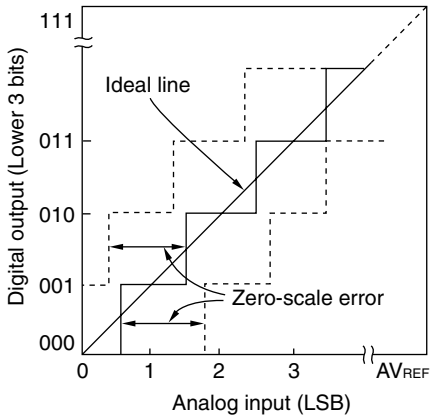



**Figure 11-41. Full-Scale Error**

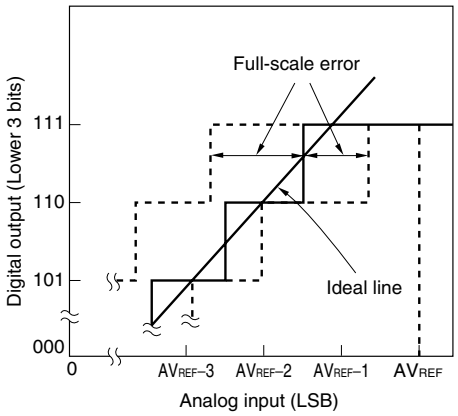



**Figure 11-42. Integral Linearity Error**

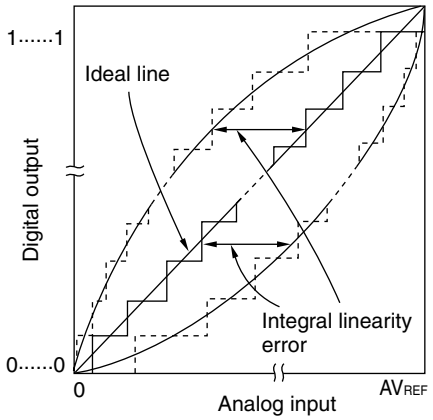
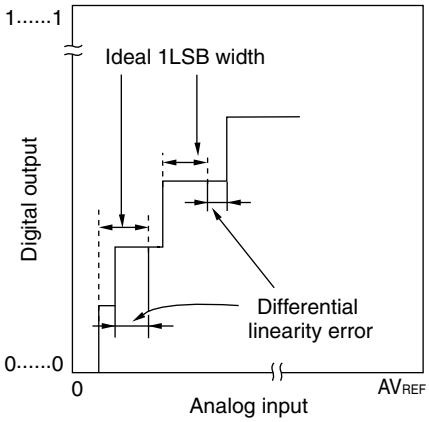



**Figure 11-43. Differential Linearity Error**



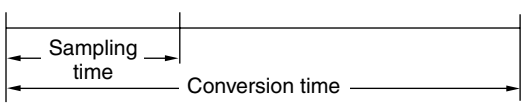

**(8) Conversion time**

This expresses the time from the start of sampling to when the digital output is obtained.

The sampling time is included in the conversion time in the characteristics table.

**(9) Sampling time**

This is the time the analog switch is turned on for the analog voltage to be sampled by the sample & hold circuit.



## 11.10 Cautions for A/D Converter

**(1) Operating current in STOP mode**

Shift to STOP mode after stopping the A/D converter (by setting bit 7 (ADCS) of A/D converter mode register 0 (ADM0) to 0). The operating current can be reduced by setting bit 0 (ADCE) of the ADM0 register to 0 at the same time.

To restart from the standby status, clear bit 0 (ADIF) of interrupt request flag register 1H (IF1H) to 0 and start operation.

**(2) Input range of ANI0 to ANI12 and ANI16 to ANI30 pins**

Observe the rated range of the ANI0 to ANI12 and ANI16 to ANI30 pins input voltage. If a voltage of $AV_{DD}$, and $AV_{REFP}$ or higher and $AV_{SS}$, and $AV_{REFM}$ or lower (even in the range of absolute maximum ratings) is input to an analog input channel, the converted value of that channel becomes undefined. In addition, the converted values of the other channels may also be affected.

When internal reference voltage (1.45 V) is selected reference voltage for the + side of the A/D converter, do not input internal reference voltage (1.45 V) or higher voltage to a pin selected by the ADS register. However, it is no problem that a pin not selected by the ADS register is input voltage greater than the internal reference voltage (1.45 V).

**Caution Internal reference voltage (1.45 V) can be used only in HS (high-speed main) mode. For detail, see Figure 24-3 Format of User Option Byte (000C2H/010C2H).**

**(3) Conflicting operations**

<1> Conflict between the A/D conversion result register (ADCR, ADCRH) write and the ADCR or ADCRH register read by instruction upon the end of conversion

The ADCR or ADCRH register read has priority. After the read operation, the new conversion result is written to the ADCR or ADCRH registers.

<2> Conflict between the ADCR or ADCRH register write and the A/D converter mode register 0 (ADM0) write, the analog input channel specification register (ADS), or A/D port configuration register (ADPC) write upon the end of conversion

The ADM0, ADS, or ADPC registers write has priority. The ADCR or ADCRH register write is not performed, nor is the conversion end interrupt request signal (INTAD) generated.

**(4) Noise countermeasures**

To maintain the 12-bit resolution, attention must be paid to noise input to the $AV_{REFP}$, $AV_{DD}$, ANI0 to ANI12, and ANI16 to ANI30 pins.

<1> Connect a capacitor with a low equivalent resistance and a good frequency response (capacitance of about <R> 0.01 $\mu$F) via the shortest possible run of relatively thick wiring to the power supply.

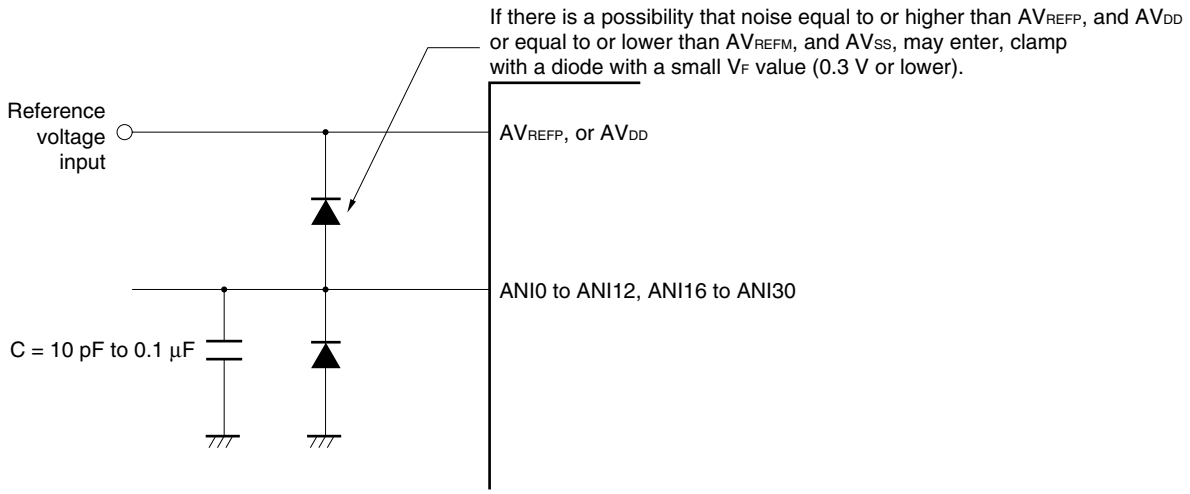
<2> The higher the output impedance of the analog input source, the greater the influence. To reduce the noise, connecting external capacitor as shown in Figure 11-44 is recommended.

<3> Do not switch these pins with other pins during conversion.

<4> The accuracy is improved if the HALT mode is set immediately after the start of conversion.

<5> Separate digital and analog signals so that they do not cross or approach each other.

**Figure 11-44. Analog Input Pin Connection**




**(5) Analog input (ANIn) pins**

<1> ANI0 to ANI12 pins (high-accuracy channel) are also used as P20 to P27, and P150 to P154 pins.
When A/D conversion is performed with any of the high-accuracy channel (ANI0 to ANI12) pins selected, do not change the output value P20 to P27, and P150 to P154 while conversion is in progress; otherwise the conversion accuracy may be degraded.

<2> If a pin adjacent to the pin whose value is being A/D converted is used as a digital I/O port pin, the A/D conversion value might differ from the expected value due to coupling noise. To prevent coupling noise, make sure that pulses whose voltage suddenly change, such as digital pulses, are not input or output to a pin adjacent to the pin whose value is being A/D converted.

**(6) Input impedance of analog input (ANIn) pins**

This A/D converter charges a sampling capacitor for sampling during sampling time.
Therefore, only a leakage current flows when sampling is not in progress, and a current that charges the capacitor flows during sampling. Consequently, the input impedance fluctuates depending on whether sampling is in progress, and on the other states.
However, in order to perform sampling accurately, the output impedance of the analog input source should be 1 kΩ or lower. If it is not possible to keep the output impedance below this level, it is recommended to either extend the sampling time or connect a capacitor of about 0.1 $\mu$F to the ANI0 to ANI12 and ANI16 to ANI30 pins. (See **Figure 11-44** for details.)
Also, if the ADCS bit is set to 0 or a reconversion is started during A/D conversion, the sampling capacitor will be insufficiently charged. This means that charging will start with an undefined conversion voltage from the next conversion in the case of setting the ADCS bit to 0, or from the current conversion in the case of starting a reconversion.
To ensure that the capacitor is fully charged, therefore, either reduce the output impedance of the analog input source or specify a sufficiently long sampling time, irrespective of the analog signal voltage variation.

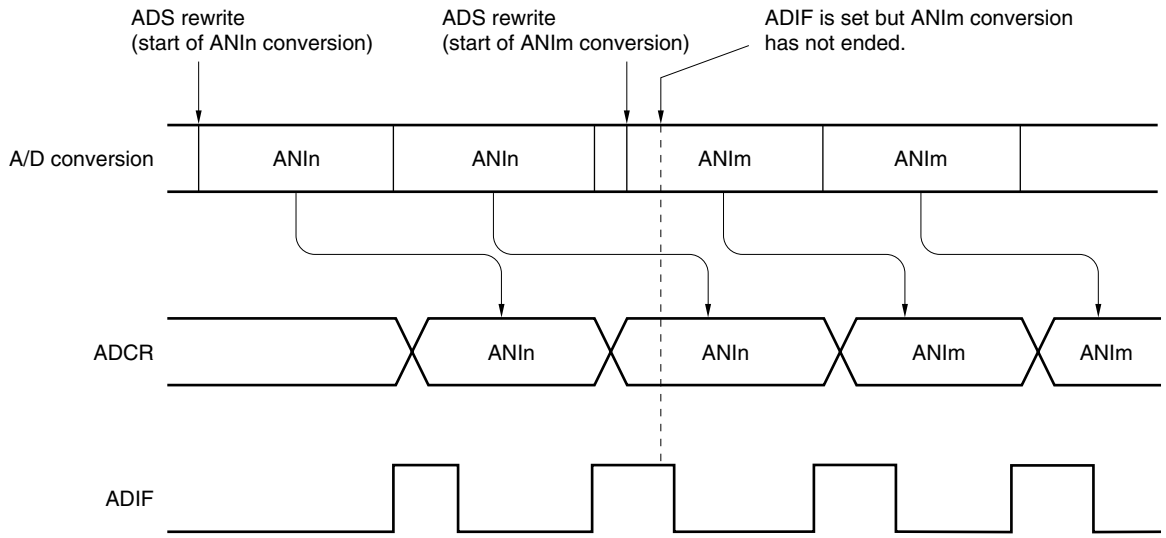
**(7) Interrupt request flag (ADIF)**

The interrupt request flag (ADIF) is not cleared even if the analog input channel specification register (ADS) is changed.

Therefore, if an analog input pin is changed during A/D conversion, the A/D conversion result and ADIF flag for the pre-change analog input may be set just before the ADS register rewrite. Caution is therefore required since, at this time, when ADIF flag is read immediately after the ADS register rewrite, ADIF flag is set despite the fact A/D conversion for the post-change analog input has not ended.

When A/D conversion is stopped and then resumed, clear ADIF flag before the A/D conversion operation is resumed.

**Figure 11-45. Timing of A/D Conversion End Interrupt Request Generation**




**(8) Conversion results just after A/D conversion start**

While in the software trigger mode or hardware trigger no-wait mode, the first A/D conversion value immediately after A/D conversion starts may not fall within the rating range if the ADCS bit is set to 1 within the stabilization wait time after the ADCE bit was set to 1. Take measures such as polling the A/D conversion end interrupt request (INTAD) and removing the first conversion result.

[Stabilization wait status]

If a high-accuracy channel is selected as the analog input channel: 0.5$\mu$s

If a test mode setting (ADTES1 bit of ADTES register = 1) is selected: 0.5$\mu$s

If a standard channel is selected as the analog input channel: 2 $\mu$s

If a temperature sensor output/internal reference voltage output are selected as the analog input channel:

(ADISS bit of ADS register = 1): 2 $\mu$s

**(9) A/D conversion result register (ADCR, ADCRH) read operation**

When a write operation is performed to A/D converter mode register 0 (ADM0), analog input channel specification register (ADS), A/D port configuration register (ADPC), and port mode control register (PMCx), the contents of the ADCR and ADCRH registers may become undefined. Read the conversion result following conversion completion before writing to the ADM0, ADS, ADPC, or PMCx register. Using a timing other than the above may cause an incorrect conversion result to be read.

**(10) Internal equivalent circuit**

The equivalent circuit of the analog input block is shown below.

**Figure 11-46. Internal Equivalent Circuit of ANIn Pin**

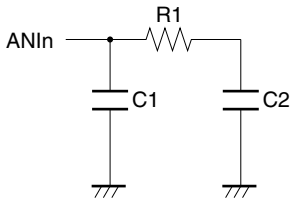



**Table 11-4. Resistance and Capacitance Values of Equivalent Circuit (Reference Values)**

| $AV_{DD}$ | ANIn pin | R1[kΩ] | C1[pF] | C2[pF] |
|---|---|---|---|---|
| 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | ANI0 to ANI12 | 7.4 | 8 | 6.3 |
| | ANI16 to ANI30 | 12.3 | 8 | 7.4 |
| 1.8 V ≤ $AV_{DD}$ ≤ 3.6 V | ANI0 to ANI12 | 11 | 8 | 6.3 |
| | ANI16 to ANI30 | 41 | 8 | 7.4 |
| 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V | ANI0 to ANI12 | 510 | 8 | 6.3 |
| | ANI16 to ANI30 | 650 | 8 | 7.4 |

**Remark** The resistance and capacitance values shown in Table 11-4 are not guaranteed values.

**(11) Starting the A/D converter**

Start the A/D converter after the $AV_{REFP}$ and $AV_{DD}$ voltages stabilize.

# CHAPTER 12 SERIAL ARRAY UNIT

A single serial array unit has up to four serial channels. Each channel can achieve 3-wire serial (CSI), UART, and simplified $I^2C$ communication.

Function assignment of each channel supported by the RL78/G1A is as shown below.

- 25-pin products

| Unit | Channel | Used as CSI | Used as UART | Used as Simplified $I^2C$ |
|---|---|---|---|---|
| 0 | 0 | CSI00 | UART0 | IIC00 |
| | 1 | – | | – |
| | 2 | – | UART1 | – |
| | 3 | CSI11 | | IIC11 |

- 32-pin products

| Unit | Channel | Used as CSI | Used as UART | Used as Simplified $I^2C$ |
|---|---|---|---|---|
| 0 | 0 | CSI00 | UART0 | IIC00 |
| | 1 | – | | – |
| | 2 | – | UART1 | – |
| | 3 | CSI11 | | IIC11 |
| 1 | 0 | CSI20 | UART2 (supporting LIN-bus) | IIC20 |
| | 1 | – | | – |

- 48-pin products

| Unit | Channel | Used as CSI | Used as UART | Used as Simplified $I^2C$ |
|---|---|---|---|---|
| 0 | 0 | CSI00 | UART0 | IIC00 |
| | 1 | CSI01 | | IIC01 |
| | 2 | – | UART1 | – |
| | 3 | CSI11 | | IIC11 |
| 1 | 0 | CSI20 | UART2 (supporting LIN-bus) | IIC20 |
| | 1 | CSI21 | | IIC21 |

- 64-pin products

| Unit | Channel | Used as CSI | Used as UART | Used as Simplified $I^2C$ |
|---|---|---|---|---|
| 0 | 0 | CSI00 | UART0 | IIC00 |
| | 1 | CSI01 | | IIC01 |
| | 2 | CSI10 | UART1 | IIC10 |
| | 3 | CSI11 | | IIC11 |
| 1 | 0 | CSI20 | UART2 (supporting LIN-bus) | IIC20 |
| | 1 | CSI21 | | IIC21 |

When “UART0” is used for channels 0 and 1 of the unit 0, CSI00 and CSI01 cannot be used, but CSI10, UART1, or IIC10 can be used.

**Caution Most of the following descriptions in this chapter use the units and channels of the 64-pin products as an example.**

## 12.1 Functions of Serial Array Unit

Each serial interface supported by the RL78/G1A has the following features.

### 12.1.1 3-wire serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21)

Data is transmitted or received in synchronization with the serial clock (SCK) output from the master channel.

3-wire serial communication is clocked communication performed by using three communication lines: one for the serial clock (SCK), one for transmitting serial data (SO), one for receiving serial data (SI).

For details about the settings, see **12.5 Operation of 3-Wire Serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) Communication.**

[Data transmission/reception]

- Data length of 7 or 8 bits
- Phase control of transmit/receive data
- MSB/LSB first selectable

[Clock control]

- Master/slave selection
- Phase control of I/O clock
- Setting of transfer period by prescaler and internal counter of each channel
- Maximum transfer rate[Note]

  During master communication: Max. $f_{CLK}$/2 (CSI00 only)
  Max. $f_{CLK}$/4
  During slave communication: Max. $f_{MCK}$/6

[Interrupt function]

- Transfer end interrupt/buffer empty interrupt

[Error detection flag]

- Overrun error

In addition, CSI00 of following channels supports the SNOOZE mode. When SCK input is detected while in the STOP mode, the SNOOZE mode makes data reception that does not require the CPU possible. Only CSI00 can be specified for asynchronous reception.

**Note** Use the clocks within a range satisfying the SCK cycle time ($t_{KCY}$) characteristics. For details, see **CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C), CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to +105°C)**.

### 12.1.2 UART (UART0 to UART2)

This is a start-stop synchronization function using two lines: serial data transmission (TxD) and serial data reception (RxD) lines. By using these two communication lines, each data frame, which consist of a start bit, data, parity bit, and stop bit, is transferred asynchronously (using the internal baud rate) between the microcontroller and the other communication party. Full-duplex UART communication can be performed by using a channel dedicated to transmission (even-numbered channel) and a channel dedicated to reception (odd-numbered channel). The LIN-bus can be implemented by using timer array unit with an external interrupt (INTP0).

For details about the settings, see **12.6 Operation of UART (UART0 to UART2) Communication.**

[Data transmission/reception]

- Data length of 7, 8, or 9 bits[Note]
- Select the MSB/LSB first
- Level setting of transmit/receive data and select of reverse
- Parity bit appending and parity check functions
- Stop bit appending

[Interrupt function]

- Transfer end interrupt/buffer empty interrupt
- Error interrupt in case of framing error, parity error, or overrun error

[Error detection flag]

- Framing error, parity error, or overrun error

In addition, UARTs of following channels supports the SNOOZE mode. When RxD input is detected while in the STOP mode, the SNOOZE mode makes data reception that does not require the CPU possible. Only UART0 can be specified for asynchronous reception.

The LIN-bus is accepted in UART2 (0 and 1 channels of unit 1) (32-pin to 64-pin products only).

[LIN-bus functions]

- Wakeup signal detection
- Break field (BF) detection
- Sync field measurement, baud rate calculation

} Using the external interrupt (INTP0) and timer array unit

**Note** Only UART0 can be specified for the 9-bit data length.

### 12.1.3 Simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21)

This is a clocked communication function to communicate with two or more devices by using two lines: serial clock (SCL) and serial data (SDA). This simplified I²C is designed for single communication with a device such as EEPROM, flash memory, or A/D converter, and therefore, it functions only as a master.

Make sure by using software, as well as operating the control registers, that the AC specifications of the start and stop conditions are observed.

For details about the settings, see **12.8 Operation of Simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21)**

[Data transmission/reception]

- Master transmission, master reception (only master function with a single master)
- ACK output function[Note] and ACK detection function
- Data length of 8 bits (When an address is transmitted, the address is specified by the higher 7 bits, and the least significant bit is used for R/W control.)
- Manual generation of start condition and stop condition

[Interrupt function]

- Transfer end interrupt

[Error detection flag]

- ACK error, or overrun error

* [Functions not supported by simplified I²C]

- Slave transmission, slave reception
- Arbitration loss detection function
- Wait detection functions

**Note** When receiving the last data, ACK will not be output if 0 is written to the SOEmn bit (serial output enable register m (SOEm)) and serial communication data output is stopped. See the processing flow in **12.8.3 (2)** for details.

**Remarks 1.** To use an I²C bus of full function, see **CHAPTER 13 SERIAL INTERFACE IICA**.

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

## 12.2 Configuration of Serial Array Unit

The serial array unit includes the following hardware.

**Table 12-1. Configuration of Serial Array Unit**

| Item | Configuration |
|---|---|
| Shift register | 8 bits or 9 bits[Note 1] |
| Buffer register | Lower 8 bits or 9 bits of serial data register mn (SDRmn)[Notes 1, 2] |
| Serial clock I/O | SCK00, SCK01, SCK10, SCK11, SCK20, SCK21 pins (for 3-wire serial I/O), SCL00, SCL01, SCL10, SCL11, SCL20, SCL21 pins (for simplified $I^2C$) |
| Serial data input | SI00, SI01, SI10, SI11, SI20, SI21 pins (for 3-wire serial I/O), RxD0, RxD1 pins (for UART), RxD2 pin (for UART supporting LIN-bus) |
| Serial data output | SO00, SO01, SO10, SO11, SO20, SO21 pins (for 3-wire serial I/O), TxD0, TxD1 pins (for UART), TxD2 pin (for UART supporting LIN-bus) |
| Serial data I/O | SDA00, SDA01, SDA10, SDA11, SDA20, SDA21 pins (for simplified $I^2C$) |
| Control registers | <Registers of unit setting block><br>• Peripheral enable register 0 (PER0)<br>• Serial clock select register m (SPSm)<br>• Serial channel enable status register m (SEm)<br>• Serial channel start register m (SSm)<br>• Serial channel stop register m (STm)<br>• Serial output enable register m (SOEm)<br>• Serial output register m (SOm)<br>• Serial output level register m (SOLm)<br>• Serial standby control register m (SSCm)<br>• Input switch control register (ISC)<br>• Noise filter enable register 0 (NFEN0) |
| | <Registers of each channel><br>• Serial data register mn (SDRmn)<br>• Serial mode register mn (SMRmn)<br>• Serial communication operation setting register mn (SCRmn)<br>• Serial status register mn (SSRmn)<br>• Serial flag clear trigger register mn (SIRmn) |
| | • Port input mode registers 0, 1 (PIM0, PIM1)<br>• Port output mode registers 0, 1, 5, 7 (POM0, POM1, POM5, POM7)<br>• Port mode control registers 0, 1, 3, 5, 7 (PMC0, PMC1, PMC3, PMC5, PMC7)<br>• Port mode registers 0, 1, 3, 5, 7 (PM0, PM1, PM3, PM5, PM7)<br>• Port registers 0, 1, 3, 5, 7 (P0, P1, P3, P5, P7) |

**Notes 1.** The number of bits used as the shift register and buffer register differs depending on the unit and channel.
- mn = 00, 01: lower 9 bits
- Other than above: lower 8 bits

**2.** The lower 8 bits of serial data register mn (SDRmn) can be read or written as the following SFR, depending on the communication mode.
- CSIp communication … SIOp (CSIp data register)
- UARTq reception … RXDq (UARTq receive data register)
- UARTq transmission … TXDq (UARTq transmit data register)
- IICr communication … SIOr (IICr data register)

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), q: UART number (q = 0 to 2), r: IIC number (r = 00, 01, 10, 11, 20, 21)

Figure 12-1 shows the block diagram of the serial array unit 0.

**Figure 12-1. Block Diagram of Serial Array Unit 0**

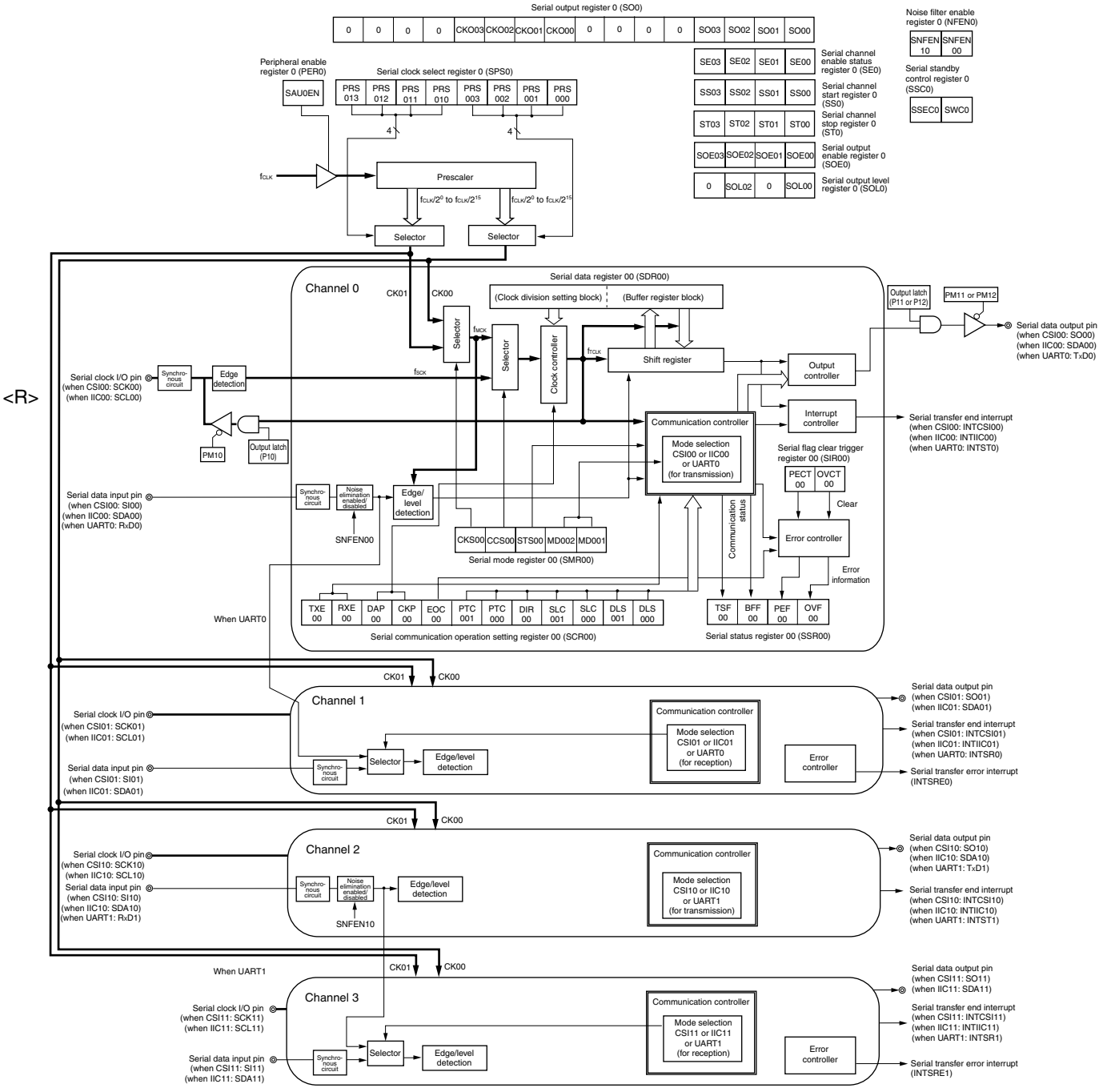
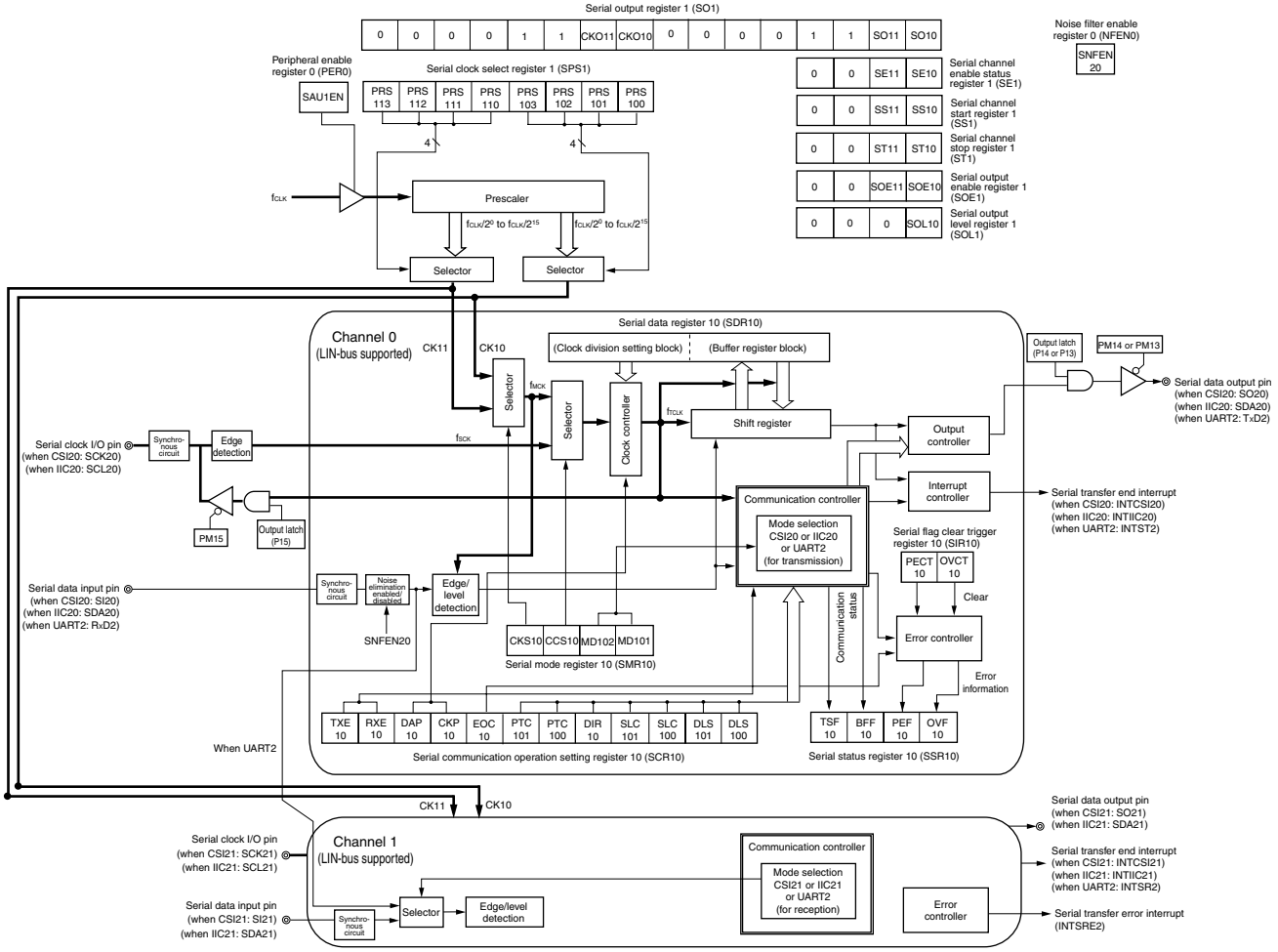


Figure 12-2 shows the block diagram of the serial array unit 1.

**Figure 12-2. Block Diagram of Serial Array Unit 1**

<R>



### 12.2.1 Shift register

This is a 9-bit register that converts parallel data into serial data or vice versa.

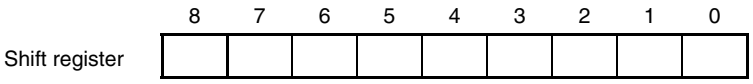
In case of the UART communication of nine bits of data, nine bits (bits 0 to 8) are used[Note 1].

The shift register cannot be directly manipulated by program.

<R> During reception, it converts data input to the serial pin into parallel data, and stores to the lower 8/9 bits of the SDRmn register.

When data is transmitted, the value transferred from the lower 8/9 bits of the SDRmn register to this register is output as serial data from the serial output pin.

<R> For details, see **12.2.2 Lower 8/9 bits of the serial data register mn (SDRmn)**.




### 12.2.2 Lower 8/9 bits of the serial data register mn (SDRmn)

The SDRmn register is the transmit/receive data register (16 bits) of channel n.

Bits 8 to 0 of SDR00, SDR01 (lower 9 bits) or bits 7 to 0 of SDR02, SDR03, SDR10[Note 1], and SDR11[Note 1] (lower 8 bits) function as a transmit/receive buffer register, and bits 15 to 9 (higher 7 bits) are used as a register that sets the division ratio of the operation clock ($f_{MCK}$).

**Remark** For the function of the higher 7 bits of the SDRmn register, see **12.3.5 Higher 7 bits of the serial data register mn (SDRmn)**.

When data is received, parallel data converted by the shift register is stored in the lower 8/9 bits. When data is to be transmitted, set transmit to be transferred to the shift register to the lower 8/9 bits.

The data stored in the lower 8/9 bits of this register is as follows, depending on the setting of bits 0 and 1 (DLSmn0, DLSmn1) of serial communication operation setting register mn (SCRmn), regardless of the output sequence of the data.

- 7-bit data length (stored in bits 0 to 6 of SDRmn register)
- 8-bit data length (stored in bits 0 to 7 of SDRmn register)
- 9-bit data length (stored in bits 0 to 8 of SDRmn register)[Note 1]

The lower 8/9 bits of the SDRmn register can be read or written[Note 2] in 8-bit units as the following SFR, depending on the communication mode.

- CSIp communication … SIOp (CSIp data register)
- UARTq reception … RXDq (UARTq receive data register)
- UARTq transmission … TXDq (UARTq transmit data register)
- IICr communication … SIOr (IICr data register)

The SDRmn register can be read or written in 16-bit units.

Reset signal generation clears the SDRmn register to 0000H.

**Notes 1.** Only following UART0 can be specified for the 9-bit data length.

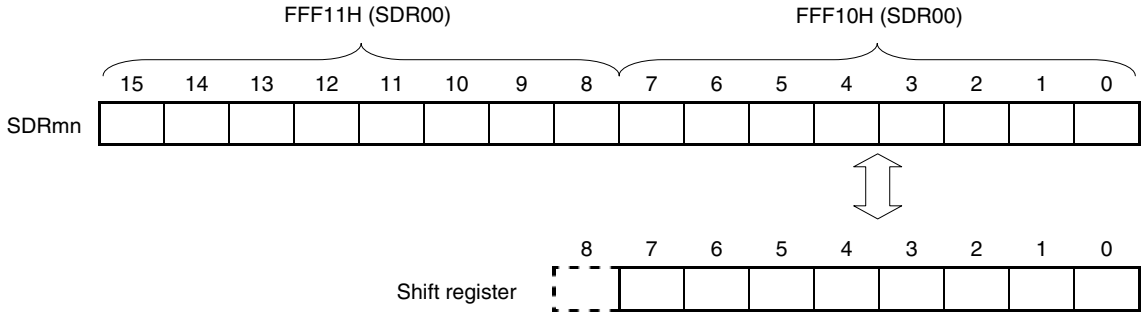
**2.** Writing in 8-bit units is prohibited when the operation is stopped (SEmn = 0).

**Remarks 1.** After data is received, “0” is stored in bits 0 to 8 in bit portions that exceed the data length.

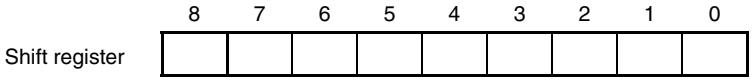
**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), q: UART number (q = 0 to 2), r: IIC number (r = 00, 01, 10, 11, 20, 21)

<R> **Figure 12-3. Format of Serial Data Register mn (SDRmn) (mn = 00, 01, 02, 03, 10, 11)**

Address: FFF10H, FFF11H (SDR00), FFF12H, FFF13H (SDR01) After reset: 0000H R/W
FFF44H, FFF45H (SDR02), FFF46H, FFF47H (SDR03),
FFF48H, FFF49H (SDR10)Note, FFF4AH, FFF4BH (SDR11)Note




• For 9-bit data communication with UART0 (mn = 00, 01)




**Note** 32, 48, 64-pin products only

**Caution For 9-bit data communication, be sure to clear bit 8 of the SDRmn register to "0".**

## 12.3 Registers Controlling Serial Array Unit

Serial array unit is controlled by the following registers.

- Peripheral enable register 0 (PER0)
- Serial clock select register m (SPSm)
- Serial mode register mn (SMRmn)
- Serial communication operation setting register mn (SCRmn)
- Serial data register mn (SDRmn)
- Serial flag clear trigger register mn (SIRmn)
- Serial status register mn (SSRmn)
- Serial channel start register m (SSm)
- Serial channel stop register m (STm)
- Serial channel enable status register m (SEm)
- Serial output enable register m (SOEm)
- Serial output level register m (SOLm)
- Serial output register m (SOm)
- Serial standby control register 0 (SSC0)
- Input switch control register (ISC)
- Noise filter enable register 0 (NFEN0)
- Port input mode registers 0, 1 (PIM0, PIM1)
- Port output mode registers 0, 1, 5, 7 (POM0, POM1, POM5, POM7)
- Port mode contorol registers 0, 1, 3, 5, 7 (PMC0, PMC1, PMC3, PMC5, PMC7)
- Port mode registers 0, 1, 3, 5, 7 (PM0, PM1, PM3, PM5, PM7)
- Port registers 0, 1, 3, 5, 7 (P0, P1, P3, P5, P7)

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

### 12.3.1 Peripheral enable register 0 (PER0)

PER0 is used to enable or disable supplying the clock to the peripheral hardware. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

When serial array unit 0 is used, be sure to set bit 2 (SAU0EN) of this register to 1.

When serial array unit 1 is used, be sure to set bit 3 (SAU1EN) of this register to 1.

The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears the PER0 register to 00H.

**Figure 12-4. Format of Peripheral Enable Register 0 (PER0)**

Address: F00F0H After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PER0 | RTCEN | 0 | ADCEN | IICA0EN | SAU1EN[Note] | SAU0EN | 0 | TAU0EN |

| SAUmEN | Control of serial array unit m input clock supply |
|---|---|
| 0 | Stops supply of input clock.<br>• SFR used by serial array unit m cannot be written.<br>• Serial array unit m is in the reset status. |
| 1 | Enables input clock supply.<br>• SFR used by serial array unit m can be read/written. |

**Note** 32, 48, and 64-pin products only.

<R> **Cautions 1. When setting serial array unit m, be sure to first set the following registers with the SAUmEN bit set to 1. If SAUmEN = 0, control registers of serial array unit m become default values and writing to them is ignored (except for the input switch control register (ISC), noise filter enable register 0 (NFEN0), port input mode registers 0, 1 (PIM0, PIM1), port output mode registers 0, 1, 5, 7 (POM0, POM1, POM5, POM7), Port mode contorol registers 0, 1, 3, 5, 7 (PMC0, PMC1, PMC3, PMC5, PMC7), port mode registers 0, 1, 3, 5, 7 (PM0, PM1, PM3, PM5, PM7), and port registers 0, 1, 3, 5, 7 (P0, P1, P3, P5, P7)).**

- **Serial clock select register m (SPSm)**
- **Serial mode register mn (SMRmn)**
- **Serial communication operation setting register mn (SCRmn)**
- **Serial data register mn (SDRmn)**
- **Serial flag clear trigger register mn (SIRmn)**
- **Serial status register mn (SSRmn)**
- **Serial channel start register m (SSm)**
- **Serial channel stop register m (STm)**
- **Serial channel enable status register m (SEm)**
- **Serial output enable register m (SOEm)**
- **Serial output level register m (SOLm)**
- **Serial output register m (SOm)**
- **Serial standby control register m (SSCm)**

**2. Be sure to clear the following bits to 0.**

**25-pin products: bits 1, 3, 6**

**32, 48, 64-pin products: bits 1, 6**

**Remark** m: Unit number (m = 0, 1)

### 12.3.2 Serial clock select register m (SPSm)

The SPSm register is a 16-bit register that is used to select two types of operation clocks (CKm0, CKm1) that are commonly supplied to each channel. CKm1 is selected by bits 7 to 4 of the SPSm register, and CKm0 is selected by bits 3 to 0.

Rewriting the SPSm register is prohibited when the register is in operation (when SEmn = 1).

The SPSm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the SPSm register can be set with an 8-bit memory manipulation instruction with SPSmL.

Reset signal generation clears the SPSm register to 0000H.

**Figure 12-5. Format of Serial Clock Select Register m (SPSm)**

Address: F0126H, F0127H (SPS0), F0166H, F0167H (SPS1)[Note 1] After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SPSm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PRS m13 | PRS m12 | PRS m11 | PRS m10 | PRS m03 | PRS m02 | PRS m01 | PRS m00 |

| PRS mk3 | PRS mk2 | PRS mk1 | PRS mk0 | Section of operation clock (CKmk)[Note 2] | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | $f_{CLK}$ = 2 MHz | $f_{CLK}$ = 5 MHz | $f_{CLK}$ = 10 MHz | $f_{CLK}$ = 20 MHz | $f_{CLK}$ = 32 MHz |
| 0 | 0 | 0 | 0 | $f_{CLK}$ | 2 MHz | 5 MHz | 10 MHz | 20 MHz | 32 MHz |
| 0 | 0 | 0 | 1 | $f_{CLK}/2$ | 1 MHz | 2.5 MHz | 5 MHz | 10 MHz | 16 MHz |
| 0 | 0 | 1 | 0 | $f_{CLK}/2^2$ | 500 kHz | 1.25 MHz | 2.5 MHz | 5 MHz | 8 MHz |
| 0 | 0 | 1 | 1 | $f_{CLK}/2^3$ | 250 kHz | 625 kHz | 1.25 MHz | 2.5 MHz | 4 MHz |
| 0 | 1 | 0 | 0 | $f_{CLK}/2^4$ | 125 kHz | 313 kHz | 625 kHz | 1.25 MHz | 2 MHz |
| 0 | 1 | 0 | 1 | $f_{CLK}/2^5$ | 62.5 kHz | 156 kHz | 313 kHz | 625 kHz | 1 MHz |
| 0 | 1 | 1 | 0 | $f_{CLK}/2^6$ | 31.3 kHz | 78.1 kHz | 156 kHz | 313 kHz | 500 kHz |
| 0 | 1 | 1 | 1 | $f_{CLK}/2^7$ | 15.6 kHz | 39.1 kHz | 78.1 kHz | 156 kHz | 250 kHz |
| 1 | 0 | 0 | 0 | $f_{CLK}/2^8$ | 7.81 kHz | 19.5 kHz | 39.1 kHz | 78.1 kHz | 125 kHz |
| 1 | 0 | 0 | 1 | $f_{CLK}/2^9$ | 3.91 kHz | 9.77 kHz | 19.5 kHz | 39.1 kHz | 62.5 kHz |
| 1 | 0 | 1 | 0 | $f_{CLK}/2^{10}$ | 1.95 kHz | 4.88 kHz | 9.77 kHz | 19.5 kHz | 31.3 kHz |
| 1 | 0 | 1 | 1 | $f_{CLK}/2^{11}$ | 977 Hz | 2.44 kHz | 4.88 kHz | 9.77 kHz | 15.6 kHz |
| 1 | 1 | 0 | 0 | $f_{CLK}/2^{12}$ | 488 Hz | 1.22 kHz | 2.44 kHz | 4.88 kHz | 7.81 kHz |
| 1 | 1 | 0 | 1 | $f_{CLK}/2^{13}$ | 244 Hz | 610 Hz | 1.22 kHz | 2.44 kHz | 3.91 kHz |
| 1 | 1 | 1 | 0 | $f_{CLK}/2^{14}$ | 122 Hz | 305 Hz | 610 Hz | 1.22 kHz | 1.95 kHz |
| 1 | 1 | 1 | 1 | $f_{CLK}/2^{15}$ | 61 Hz | 153 kHz | 305 Hz | 610 Hz | 977 Hz |

<R> **Notes 1.** 32, 48, and 64-pin products

**2.** When changing the clock selected for $f_{CLK}$ (by changing the system clock control register (CKC) value), do so after having stopped (serial channel stop register m (STm) = 000FH) the operation of the serial array unit (SAU).

**Caution Be sure to clear bits 15 to 8 to "0".**

**Remarks 1.** $f_{CLK}$: CPU/peripheral hardware clock frequency

**2.** m: Unit number (m = 0, 1)

**3.** k = 0, 1

### 12.3.3 Serial mode register mn (SMRmn)

The SMRmn register is a register that sets an operation mode of channel n. It is also used to select an operation clock ($f_{MCK}$), specify whether the serial clock ($f_{SCK}$) may be input or not, set a start trigger, an operation mode (CSI, UART, or simplified $I^2C$), and an interrupt source. This register is also used to invert the level of the receive data only in the UART mode.

Rewriting the SMRmn register is prohibited when the register is in operation (when SEmn = 1). However, the MDmn0 bit can be rewritten during operation.

The SMRmn register can be set by a 16-bit memory manipulation instruction.

Reset signal generation sets the SMRmn register to 0020H.

**Figure 12-6. Format of Serial Mode Register mn (SMRmn) (1/2)**

Address: F0110H, F0111H (SMR00) to F0116H, F0117H (SMR03), F0150H, F0151H (SMR10), F0152H, F0153H (SMR11)[Note 1] After reset: 0020H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SMRmn | CKS mn | CCS mn | 0 | 0 | 0 | 0 | 0 | STS mn Note 2 | 0 | SIS mn0 Note 2 | 1 | 0 | 0 | MD mn2 | MD mn1 | MD mn0 |

| CKS mn | Selection of operation clock ($f_{MCK}$) of channel n |
|---|---|
| 0 | Operation clock CKm0 set by the SPSm register |
| 1 | Operation clock CKm1 set by the SPSm register |
| Operation clock ($f_{MCK}$) is used by the edge detector. In addition, depending on the setting of the CCSmn bit and the higher 7 bits of the SDRmn register, a transfer clock ($f_{TCLK}$) is generated. | |

| CCS mn | Selection of transfer clock ($f_{TCLK}$) of channel n |
|---|---|
| 0 | Divided operation clock $f_{MCK}$ specified by the CKSmn bit |
| 1 | Clock input $f_{SCK}$ from the SCKp pin (slave transfer in CSI mode) |
| Transfer clock $f_{TCLK}$ is used for the shift register, communication controller, output controller, interrupt controller, and error controller. When CCSmn = 0, the division ratio of operation clock ($f_{MCK}$) is set by the higher 7 bits of the SDRmn register. | |

| STS mn Note 2 | Selection of start trigger source |
|---|---|
| 0 | Only software trigger is valid (selected for CSI, UART transmission, and simplified $I^2C$). |
| 1 | Valid edge of the RxDq pin (selected for UART reception) |
| Transfer is started when the above source is satisfied after 1 is set to the SSm register. | |

<R> **Notes 1.** SMR00 to SMR03: All products
SMR10, SMR11: 32, 48, and 64-pin products

**2.** The SMR01, SMR03, and SMR11 registers only.

**Caution Be sure to clear bits 13 to 9, 7, 4, and 3 (or bits 13 to 6, 4, and 3 for the SMR00, SMR02, or SMR10 register) to "0". Be sure to set bit 5 to "1".**

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), q: UART number (q = 0 to 2), r: IIC number (r = 00, 01, 10, 11, 20, 21)

**Figure 12-6. Format of Serial Mode Register mn (SMRmn) (2/2)**

Address: F0110H, F0111H (SMR00) to F0116H, F0117H (SMR03), F0150H, F0151H (SMR10), F0152H, F0153H (SMR11)[Note 1] After reset: 0020H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SMRmn | CKS mn | CCS mn | 0 | 0 | 0 | 0 | 0 | STS mn Note 2 | 0 | SIS mn0 Note 2 | 1 | 0 | 0 | MD mn2 | MD mn1 | MD mn0 |

| SIS mn0 Note 2 | Controls inversion of level of receive data of channel n in UART mode |
|---|---|
| 0 | Falling edge is detected as the start bit.<br>The input communication data is captured as is. |
| 1 | Rising edge is detected as the start bit.<br>The input communication data is inverted and captured. |

| MD mn2 | MD mn1 | Setting of operation mode of channel n |
|---|---|---|
| 0 | 0 | CSI mode |
| 0 | 1 | UART mode |
| 1 | 0 | Simplified $I^2C$ mode |
| 1 | 1 | Setting prohibited |

| MD mn0 | Selection of interrupt source of channel n |
|---|---|
| 0 | Transfer end interrupt |
| 1 | Buffer empty interrupt<br>(Occurs when data is transferred from the SDRmn register to the shift register.) |
| For successive transmission, the next transmit data is written by setting the MDmn0 bit to 1 when SDRmn data has run out. | |

<R> **Notes 1.** SMR00 to SMR03: All products
SMR10, SMR11: 32, 48, and 64-pin products

**2.** The SMR01, SMR03, and SMR11 registers only.

**Caution Be sure to clear bits 13 to 9, 7, 4, and 3 (or bits 13 to 6, 4, and 3 for the SMR00, SMR02, or SMR10 register) to "0". Be sure to set bit 5 to "1".**

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), q: UART number (q = 0 to 2), r: IIC number (r = 00, 01, 10, 11, 20, 21)

### 12.3.4 Serial communication operation setting register mn (SCRmn)

The SCRmn register is a communication operation setting register of channel n. It is used to set a data transmission/reception mode, phase of data and clock, whether an error signal is to be masked or not, parity bit, start bit, stop bit, and data length.

Rewriting the SCRmn register is prohibited when the register is in operation (when SEmn = 1).

The SCRmn register can be set by a 16-bit memory manipulation instruction.

Reset signal generation sets the SCRmn register to 0087H.

**Figure 12-7. Format of Serial Communication Operation Setting Register mn (SCRmn) (1/2)**

Address: F0118H, F0119H (SCR00) to F011EH, F011FH (SCR03), After reset: 0087H R/W
F0158H, F0159H (SCR10), F015AH, F015BH (SCR11)[Note 1]

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SCRmn | TXE mn | RXE mn | DAP mn | CKP mn | 0 | EOC mn | PTC mn1 | PTC mn0 | DIR mn | 0 | SLCm n1[Note 2] | SLC mn0 | 0 | 1 | DLSm n1[Note 3] | DLS mn0 |

| TXE mn | RXE mn | Setting of operation mode of channel n |
|---|---|---|
| 0 | 0 | Disable communication. |
| 0 | 1 | Reception only |
| 1 | 0 | Transmission only |
| 1 | 1 | Transmission/reception |

| | DAP mn | CKP mn | Selection of data and clock phase in CSI mode | Type |
|---|---|---|---|---|
| <R> | 0 | 0 | SCKp<br>SOp D7 D6 D5 D4 D3 D2 D1 D0<br>Sp input timing | 1 |
| <R> | 0 | 1 | SCKp<br>SOp D7 D6 D5 D4 D3 D2 D1 D0<br>Sp input timing | 2 |
| <R> | 1 | 0 | SCKp<br>SOp D7 D6 D5 D4 D3 D2 D1 D0<br>Sp input timing | 3 |
| <R> | 1 | 1 | SCKp<br>SOp D7 D6 D5 D4 D3 D2 D1 D0<br>Sp input timing | 4 |
| Be sure to set DAPmn, CKPmn = 0, 0 in the UART mode and simplified $I^2C$ mode. | | | | |

| EOC mn | Mask control of error interrupt signal (INTSREx (x = 0 to 2)) |
|---|---|
| 0 | Disables generation of error interrupt INTSREx (INTSRx is generated). |
| 1 | Enables generation of error interrupt INTSREx (INTSRx is not generated if an error occurs). |
| Set EOCmn = 0 in the CSI mode, simplified $I^2C$ mode, and during UART transmission[Note 4]. | |

<R> **Notes 1.** SCR00 to SCR03: All products
SCR10, SCR11: 32, 48, and 64-pin products

**2.** The SCR00, SCR02, and SCR10 registers only.

**3.** The SCR00 and SCR01 registers only. Others are fixed to 1.

**4.** When using CSImn not with EOCmn = 0, error interrupt INTSREn may be generated.

**Caution Be sure to clear bits 3, 6, and 11 to "0" (Also clear bit 5 of the SCR01, SCR03, or SCR11 register to 0). Be sure to set bit 2 to "1".**

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21)

**Figure 12-7. Format of Serial Communication Operation Setting Register mn (SCRmn) (2/2)**

Address: F0118H, F0119H (SCR00) to F011EH, F011FH (SCR03), F0158H, F0159H (SCR10), F015AH, F015BH (SCR11)[Note 1] After reset: 0087H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SCRmn | TXEmn | RXEmn | DAPmn | CKPmn | 0 | EOCmn | PTCmn1 | PTCmn0 | DIRmn | 0 | SLCmn1[Note 2] | SLCmn0 | 0 | 1 | DLSmn1[Note 3] | DLSmn0 |

| PTCmn1 | PTCmn0 | Setting of parity bit in UART mode | |
|---|---|---|---|
| | | Transmission | Reception |
| 0 | 0 | Does not output the parity bit. | Receives without parity |
| 0 | 1 | Outputs 0 parity[Note 4]. | No parity judgment |
| 1 | 0 | Outputs even parity. | Judged as even parity. |
| 1 | 1 | Outputs odd parity. | Judges as odd parity. |
| Be sure to set PTCmn1, PTCmn0 = 0, 0 in the CSI mode and simplified $I^2C$ mode. | | | |

| DIRmn | Selection of data transfer sequence in CSI and UART modes |
|---|---|
| 0 | Inputs/outputs data with MSB first. |
| 1 | Inputs/outputs data with LSB first. |
| Be sure to clear DIRmn = 0 in the simplified $I^2C$ mode. | |

| SLCmn1[Note 2] | SLCmn0 | Setting of stop bit in UART mode |
|---|---|---|
| 0 | 0 | No stop bit |
| 0 | 1 | Stop bit length = 1 bit |
| 1 | 0 | Stop bit length = 2 bits (mn = 00, 02, 10 only) |
| 1 | 1 | Setting prohibited |
| When the transfer end interrupt is selected, the interrupt is generated when all stop bits have been completely transferred.<br>Set 1 bit (SLCmn1, SLCmn0 = 0, 1) during UART reception and in the simplified $I^2C$ mode.<br>Set no stop bit (SLCmn1, SLCmn0 = 0, 0) in the CSI mode.<br>Set 1 bit (SLCmn1, SLCmn0 = 0, 1) or 2 bits (SLCmn1, SLCmn0 = 1, 0) during UART transmission. | | |

| DLSmn1[Note 3] | DLSmn0 | Setting of data length in CSI and UART modes |
|---|---|---|
| 0 | 1 | 9-bit data length (stored in bits 0 to 8 of the SDRmn register) (settable in UART mode only) |
| 1 | 0 | 7-bit data length (stored in bits 0 to 6 of the SDRmn register) |
| 1 | 1 | 8-bit data length (stored in bits 0 to 7 of the SDRmn register) |
| Other than above | | Setting prohibited |
| Be sure to set DLSmn1, DLSmn0 = 1, 1 in the simplified $I^2C$ mode. | | |

<R> **Notes 1.** SCR00 to SCR03: All products
SCR10, SCR11: 32, 48, and 64-pin products

**2.** The SCR00, SCR02, and SCR10 registers only.

**3.** The SCR00 and SCR01 registers only. Others are fixed to 1.

**4.** 0 is always added regardless of the data contents.

(**Caution** and **remark** are listed on the next page.)

**Caution Be sure to clear bits 3, 6, and 11 to "0". (Also clear bit 5 of the SCR01, SCR03, or SCR11 register to 0). Be sure to set bit 2 to "1".**

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21)

<R> **12.3.5 Higher 7 bits of the serial data register mn (SDRmn)**

The SDRmn register is the transmit/receive data register (16 bits) of channel n.

<R> Bits 8 to 0 (lower 9 bits) of SDR00, SDR01, SDR10, SDR11 or bits 7 to 0 (lower 8 bits) of SDR02, SDR03, SDR10[Note 1], and SDR11[Note 1] function as a transmit/receive buffer register, and bits 15 to 9 (higher 7 bits) are used as a register that sets the division ratio of the operation clock ($f_{MCK}$).

<R> **Remark** For the function of the lower 8/9 bits of the SDRmn register, see **12.2.2 Lower 8/9 bits of the serial data register mn (SDRmn)**.

If the CCSmn bit of serial mode register mn (SMRmn) is cleared to 0, the clock set by dividing the operating clock by the bits 15 to 9 (higher 7 bits) of the SDRmn register is used as the transfer clock.

If the CCSmn bit of serial mode register mn (SMRmn) is set to 1, bits 15 to 9 of SDR01, SDR00 (higher 7 bits) are set to "0000000B". $f_{SCK}$ of input clock from the SCKp pin (slave transfer of CSI mode) is the transfer clock.

The higher 7 bits can be written or read only when the operation is stopped (SEmn = 0). During operation (SEmn = 1), a value is written only to the lower 8/9 bits of the SDRmn register. When the SDRmn register is read during operation, the higher 7 bits are always read as 0.

The SDRmn register can be read or written in 16-bit units.

Reset signal generation clears the SDRmn register to 0000H.

**Figure 12-8. Format of Serial Data Register mn (SDRmn)**

Address: FFF10H, FFF11H (SDR00), FFF12H, FFF13H (SDR01) After reset: 0000H R/W

FFF11H (SDR00)

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SDRmn | | | | | | | | | | | | | | | | |

Address: FFF44H, FFF45H (SDR02), FFF46H, FFF47H (SDR03), FFF48H, FFF49H (SDR10)Note, FFF4AH, FFF4BH (SDR11)Note After reset: 0000H R/W

FFF45H (SDR02) FFF44H (SDR02)

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SDRmn | | | | | | | | 0 | | | | | | | | |

| SDRmn[15:9] | | | | | | | Transfer clock setting by dividing the operating clock ($f_{MCK}$) |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | $f_{MCK}/2$ |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | $f_{MCK}/4$ |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | $f_{MCK}/6$ |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | $f_{MCK}/8$ |
| • | • | • | • | • | • | • | • |
| • | • | • | • | • | • | • | • |
| • | • | • | • | • | • | • | • |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 | $f_{MCK}/254$ |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | $f_{MCK}/256$ |

<R> **Note** 32, 48, and 64-pin products

**Cautions 1. Be sure to clear bit 8 of the SDR02, SDR03, SDR10, and SDR11 to "0".**

<R> **2. Setting SDRmn[15:9] = (0000000B, 0000001B) is prohibited when UART is used.**

**3. Setting SDRmn[15:9] = 0000000B is prohibited when simplified $I^2C$ is used. Set SDRmn[15:9] to 0000001B or greater.**

**4. Do not write eight bits to the lower eight bits if operation is stopped (SEmn = 0). (If these bits are written to, the higher seven bits are cleared to 0.)**

<R> **Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

### 12.3.6 Serial flag clear trigger register mn (SIRmn)

The SIRmn register is a trigger register that is used to clear each error flag of channel n.

When each bit (FECTmn, PECTmn, OVCTmn) of this register is set to 1, the corresponding bit (FEFmn, PEFmn, OVFmn) of serial status register mn (SSRmn) is cleared to 0. Because the SIRmn register is a trigger register, it is cleared immediately when the corresponding bit of the SSRmn register is cleared.

The SIRmn register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the SIRmn register can be set with an 8-bit memory manipulation instruction with SIRmnL.

Reset signal generation clears the SIRmn register to 0000H.

**Figure 12-9. Format of Serial Flag Clear Trigger Register mn (SIRmn)**

Address: F0108H, F0109H (SIR00) to F010EH, F010FH (SIR03), F0148H, F0149H (SIR10),F014AH, F014BH (SIR11)[Note 1] After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SIRmn | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | FECTmn [Note 2] | PECTmn | OVCTmn |

| FECTmn | Clear trigger of framing error of channel n |
|---|---|
| 0 | Not cleared |
| 1 | Clears the FEFmn bit of the SSRmn register to 0. |

| PECTmn | Clear trigger of parity error flag of channel n |
|---|---|
| 0 | Not cleared |
| 1 | Clears the PEFmn bit of the SSRmn register to 0. |

| OVCTmn | Clear trigger of overrun error flag of channel n |
|---|---|
| 0 | Not cleared |
| 1 | Clears the OVFmn bit of the SSRmn register to 0. |

<R> **Notes 1.** SIR00 to SIR03: All products
SIR10, SIR11: 32, 48, and 64-pin products

**2.** The SIR01, SIR03, and SIR11 registers only.

**Caution Be sure to clear bits 15 to 3 (or bits 15 to 2 for the SIR00, SIR02, or SIR10 register) to "0".**

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

**2.** When the SIRmn register is read, 0000H is always read.

### 12.3.7 Serial status register mn (SSRmn)

The SSRmn register is a register that indicates the communication status and error occurrence status of channel n. The errors indicated by this register are a framing error, parity error, and overrun error.

The SSRmn register can be read by a 16-bit memory manipulation instruction.

The lower 8 bits of the SSRmn register can be set with an 8-bit memory manipulation instruction with SSRmnL.

Reset signal generation clears the SSRmn register to 0000H.

**Figure 12-10. Format of Serial Status Register mn (SSRmn) (1/2)**

Address: F0100H, F0101H (SSR00) to F0106H, F0107H (SSR03), F0140H, F0141H (SSR10), F0142H, F0143H (SSR11)[Note 1] After reset: 0000H R

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SSRmn | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | TSF mn | BFF mn | 0 | 0 | FEFm n[Note 2] | PEF mn | OVF mn |

| TSF mn | Communication status indication flag of channel n |
|---|---|
| 0 | Communication is stopped or suspended. |
| 1 | Communication is in progress. |
| <Clear conditions><br>• The STmn bit of the STm register is set to 1 (communication is stopped) or the SSmn bit of the SSm register is set to 1 (communication is suspended).<br>• Communication ends.<br><Set condition><br>• Communication starts. | |

| BFF mn | Buffer register status indication flag of channel n |
|---|---|
| 0 | Valid data is not stored in the SDRmn register. |
| 1 | Valid data is stored in the SDRmn register. |
| <Clear conditions><br>• Transferring transmit data from the SDRmn register to the shift register ends during transmission.<br>• Reading receive data from the SDRmn register ends during reception.<br>• The STmn bit of the STm register is set to 1 (communication is stopped) or the SSmn bit of the SSm register is set to 1 (communication is enabled).<br><Set conditions><br>• Transmit data is written to the SDRmn register while the TXEmn bit of the SCRmn register is set to 1 (transmission or transmission and reception mode in each communication mode).<br>• Receive data is stored in the SDRmn register while the RXEmn bit of the SCRmn register is set to 1 (reception or transmission and reception mode in each communication mode).<br>• A reception error occurs. | |

<R> **Notes 1.** SSR00 to SSR03: All products
SSR10, SSR11: 32, 48, and 64-pin products

**2.** The SSR01, SSR03, and SSR11 registers only.

**Caution When the CSI is performing reception operations in the SNOOZE mode (SWCm = 1), the BFFmn flag will not change.**

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

**Figure 12-10. Format of Serial Status Register mn (SSRmn) (2/2)**

Address: F0100H, F0101H (SSR00) to F0106H, F0107H (SSR03), F0140H, F0141H (SSR10), F0142H, F0143H (SSR11)[Note 1] After reset: 0000H R

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SSRmn | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | TSFmn | BFFmn | 0 | 0 | FEFmn[Note 2] | PEFmn | OVFmn |

| FEFmn[Note 2] | Framing error detection flag of channel n |
|---|---|
| 0 | No error occurs. |
| 1 | An error occurs (during UART reception). |
| <Clear condition><br>• 1 is written to the FECTmn bit of the SIRmn register.<br><Set condition><br>• A stop bit is not detected when UART reception ends. | |

| PEFmn | Parity/ACK error detection flag of channel n |
|---|---|
| 0 | No error occurs. |
| 1 | Parity error occurs (during UART reception) or ACK is not detected (during $I^2C$ transmission). |
| <Clear condition><br>• 1 is written to the PECTmn bit of the SIRmn register.<br><Set condition><br>• The parity of the transmit data and the parity bit do not match when UART reception ends (parity error).<br>• No ACK signal is returned from the slave channel at the ACK reception timing during $I^2C$ transmission (ACK is not detected). | |

| OVFmn | Overrun error detection flag of channel n |
|---|---|
| 0 | No error occurs. |
| 1 | An error occurs |
| <Clear condition><br>• 1 is written to the OVCTmn bit of the SIRmn register.<br><Set condition><br>• Even though receive data is stored in the SDRmn register, that data is not read and transmit data or the next receive data is written while the RXEmn bit of the SCRmn register is set to 1 (reception or transmission and reception mode in each communication mode).<br>• Transmit data is not ready for slave transmission or transmission and reception in CSI mode. | |

<R> **Notes 1.** SSR00 to SSR03: All products
SSR10, SSR11: 32, 48, and 64-pin products
**2.** The SSR01, SSR03, and SSR11 registers only.

**Cautions 1. If data is written to the SDRmn register when BFFmn = 1, the transmit/receive data stored in the register is discarded and an overrun error (OVEmn = 1) is detected.**
**2. When the CSI is performing reception operations in the SNOOZE mode (SWCm = 1), the OVFmn flag will not change.**

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

### 12.3.8 Serial channel start register m (SSm)

The SSm register is a trigger register that is used to enable starting communication/count by each channel.

When 1 is written a bit of this register (SSmn), the corresponding bit (SEmn) of serial channel enable status register m (SEm) is set to 1 (Operation is enabled). Because the SSmn bit is a trigger bit, it is cleared immediately when SEmn = 1.

The SSm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the SSm register can be set with an 1-bit or 8-bit memory manipulation instruction with SSmL.

Reset signal generation clears the SSm register to 0000H.

**Figure 12-11. Format of Serial Channel Start Register m (SSm)**

Address: F0122H, F0123H (SS0) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SS0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SS03 | SS02 | SS01 | SS00 |

Address: F0162H, F0163H (SS1)[Note 1] After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SS1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SS11 | SS10 |

| SSmn | Operation start trigger of channel n |
|---|---|
| 0 | No trigger operation |
| 1 | Sets the SEmn bit to 1 and enters the communication wait status[Note 2]. |

**Notes 1.** 32, 48, 64-pin products only

**2.** If set the SSmn = 1 to during a communication operation, will wait status to stop the communication. At this time, holding status value of control register and shift register, SCKmn and SOmn pins, and FEFmn, PEFmn, OVFmn flags.

**Cautions 1. Be sure to clear bits 15 to 4 of the SS0 register and bits 15 to 2 of the SS1 register to "0".**

**2. For the UART reception, set the RXEmn bit of SCRmn register to 1, and then be sure to set SSmn to 1 after 4 or more $f_{MCK}$ clocks have elapsed.**

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

**2.** When the SSm register is read, 0000H is always read.

### 12.3.9 Serial channel stop register m (STm)

The STm register is a trigger register that is used to enable stopping communication/count by each channel.

When 1 is written a bit of this register (STmn), the corresponding bit (SEmn) of serial channel enable status register m (SEm) is cleared to 0 (operation is stopped). Because the STmn bit is a trigger bit, it is cleared immediately when SEmn = 0.

The STm register can set written by a 16-bit memory manipulation instruction.

The lower 8 bits of the STm register can be set with a 1-bit or 8-bit memory manipulation instruction with STmL.

Reset signal generation clears the STm register to 0000H.

**Figure 12-12. Format of Serial Channel Stop Register m (STm)**

Address: F0124H, F0125H (ST0) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ST0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ST03 | ST02 | ST01 | ST00 |

Address: F0164H, F0165H (ST1)[Note 1] After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ST1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ST11 | ST10 |

| STmn | Operation stop trigger of channel n |
|---|---|
| 0 | No trigger operation |
| 1 | Clears the SEmn bit to 0 and stops the communication operation[Note 2]. |

**Notes 1.** 32, 48, 64-pin products only

**2.** Holding status value of the control register and shift register, the SCKmn and SOmn pins, and FEFmn, PEFmn, OVFmn flags.

**Caution Be sure to clear bits 15 to 4 of the ST0 register and bits 15 to 2 of the ST1 register to "0".**

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

**2.** When the STm register is read, 0000H is always read.

### 12.3.10 Serial channel enable status register m (SEm)

The SEm register indicates whether data transmission/reception operation of each channel is enabled or stopped.

When 1 is written a bit of serial channel start register m (SSm), the corresponding bit of this register is set to 1. When 1 is written a bit of serial channel stop register m (STm), the corresponding bit is cleared to 0.

Channel n that is enabled to operate cannot rewrite by software the value of the CKOmn bit (serial clock output of channel n) of serial output register m (SOm) to be described below, and a value reflected by a communication operation is output from the serial clock pin.

Channel n that stops operation can set the value of the CKOmn bit of the SOm register by software and output its value from the serial clock pin. In this way, any waveform, such as that of a start condition/stop condition, can be created by software.

The SEm register can be read by a 16-bit memory manipulation instruction.

The lower 8 bits of the SEm register can be set with a 1-bit or 8-bit memory manipulation instruction with SEmL.

Reset signal generation clears the SEm register to 0000H.

**Figure 12-13. Format of Serial Channel Enable Status Register m (SEm)**

Address: F0120H, F0121H (SE0) After reset: 0000H R

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SE0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SE03 | SE02 | SE01 | SE00 |

Address: F0160H, F0161H (SE1)[Note] After reset: 0000H R

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SE1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SE11 | SE10 |

| SEmn | Indication of operation enable/stop status of channel n |
|---|---|
| 0 | Operation stops |
| 1 | Operation is enabled. |

**Note** 32, 48, 64-pin products only

**Caution Be sure to clear bits 15 to 4 of the SE0 register and bits 15 to 2 of the SE1 register to "0".**

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

### 12.3.11 Serial output enable register m (SOEm)

The SOEm register is a register that is used to enable or stop output of the serial communication operation of each channel.

Channel n that enables serial output cannot rewrite by software the value of the SOmn bit of serial output register m (SOm) to be described below, and a value reflected by a communication operation is output from the serial data output pin.

For channel n, whose serial output is stopped, the SOmn bit value of the SOm register can be set by software, and that value can be output from the serial data output pin. In this way, any waveform of the start condition and stop condition can be created by software.

The SOEm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the SOEm register can be set with a 1-bit or 8-bit memory manipulation instruction with SOEmL.

Reset signal generation clears the SOEm register to 0000H.

**Figure 12-14. Format of Serial Output Enable Register m (SOEm)**

Address: F012AH, F012BH (SOE0) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SOE0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SOE03 | SOE02 | SOE01 | SOE00 |

Address: F016AH, F016BH (SOE1)[Note] After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SOE1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SOE11 | SOE10 |

| SOEmn | Serial output enable/stop of channel n |
|---|---|
| 0 | Stops output by serial communication operation. |
| 1 | Enables output by serial communication operation. |

**Note** 32, 48, 64-pin products only

**Caution Be sure to clear bits 15 to 4 of the SOE0 register, and bits 15 to 2 of the SOE1 register to "0".**

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

### 12.3.12 Serial output register m (SOm)

The SOm register is a buffer register for serial output of each channel.

The value of the SOmn bit of this register is output from the serial data output pin of channel n.

The value of the CKOmn bit of this register is output from the serial clock output pin of channel n.

The SOmn bit of this register can be rewritten by software only when serial output is disabled (SOEmn = 0). When serial output is enabled (SOEmn = 1), rewriting by software is ignored, and the value of the register can be changed only by a serial communication operation.

The CKOmn bit of this register can be rewritten by software only when the channel operation is stopped (SEmn = 0). While channel operation is enabled (SEmn = 1), rewriting by software is ignored, and the value of the CKOmn bit can be changed only by a serial communication operation.

To use the pin for serial interface as a port function pin, set the corresponding CKOmn and SOmn bits to “1”.

The SOm register can be set by a 16-bit memory manipulation instruction.

<R> Reset signal generation clears the SO0 register to 0F0FH, and sets the SO1 register to 0303H.

**Figure 12-15. Format of Serial Output Register m (SOm)**

Address: F0128H, F0129H (SO0) After reset: 0F0FH R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SO0 | 0 | 0 | 0 | 0 | CKO 03 | CKO 02 | CKO 01 | CKO 00 | 0 | 0 | 0 | 0 | SO 03 | SO 02 | SO 01 | SO 00 |

<R> Address: F0168H, F0169H (SO1)[Note] After reset: 0303H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SO1 | 0 | 0 | 0 | 0 | 0 | 0 | CKO 11 | CKO 10 | 0 | 0 | 0 | 0 | 0 | 0 | SO 11 | SO 10 |

| CKO mn | Serial clock output of channel n |
|---|---|
| 0 | Serial clock output value is “0”. |
| 1 | Serial clock output value is “1”. |

| SO mn | Serial data output of channel n |
|---|---|
| 0 | Serial data output value is “0”. |
| 1 | Serial data output value is “1”. |

**Note** 32, 48, 64-pin products only

**Caution Be sure to clear bits 15 to 12, 7 to 4 of the SO0 register, and bits 15 to 10, 7 to 2 of the SO1 register to “0”.**

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3)

### 12.3.13 Serial output level register m (SOLm)

The SOLm register is a register that is used to set inversion of the data output level of each channel.

This register can be set only in the UART mode. Be sure to set 0 for corresponding bit in the CSI mode and simplifies $I^2C$ mode.

Inverting channel n by using this register is reflected on pin output only when serial output is enabled (SOEmn = 1). When serial output is disabled (SOEmn = 0), the value of the SOmn bit is output as is.

Rewriting the SOLm register is prohibited when the register is in operation (when SEmn = 1).

The SOLm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the SOLm register can be set with an 8-bit memory manipulation instruction with SOLmL.

Reset signal generation clears the SOLm register to 0000H.

**Figure 12-16. Format of Serial Output Level Register m (SOLm)**

Address: F0134H, F0135H (SOL0) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SOL0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SOL02 | 0 | SOL00 |

Address: F0174H, F0175H (SOL1)[Note] After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SOL1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SOL10 |

| SOLmn | Selects inversion of the level of the transmit data of channel n in UART mode |
|---|---|
| 0 | Communication data is output as is. |
| 1 | Communication data is inverted and output. |

**Note** 32, 48, 64-pin products only

**Caution Be sure to clear bits 15 to 3, and 1 of the SOL0 register and bits 15 to 1 of the SOL1 register to "0".**

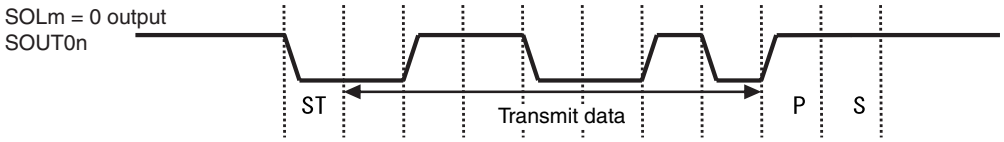
(**Remark** is listed on the next page.)

Figure 12-17 shows examples in which the level of transmit data is reversed during UART transmission.

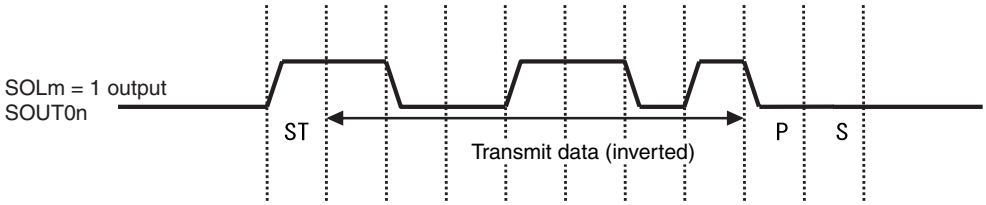
<R>

**Figure 12-17. Examples of Reverse Transmit Data**

**(a) Non-reverse Output (SOLmn = 0)**




**(b) Reverse Output (SOLmn = 1)**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0, 2)

### 12.3.14 Serial standby control register 0 (SSC0)

The SSC0 register is used to control the startup of reception (the SNOOZE mode) while in the STOP mode when receiving CSI00 or UART0 serial data.

The SSC0 register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the SSC0 register can be set with an 8-bit memory manipulation instruction with SSC0L.

Reset signal generation clears the SSC0 register to 0000H.

**Caution The maximum transfer rate in the SNOOZE mode is as follows.**

- **When using CSI00: Until 1 Mbps**
- **When using UART0: 4800 bps only**

**Figure 12-18. Format of Serial Standby Control Register 0 (SSC0)**

Address: F0138H (SSC0) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SSC0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SSEC0 | SWC0 |

<R>

| SSEC0 | Selection of whether to enable or stop the generation of communication error interrupts in the SNOOZE mode |
|---|---|
| 0 | Enable the generation of error interrupts (INTSRE0). |
| 1 | Stop the generation of error interrupts (INTSRE0). |
| • The SSEC0 bit can be set to 1 or 0 only when both the SWC0 and EOC0n bits are set to 1 during UART reception in the SNOOZE mode. In other cases, clear the SSEC0 bit to 0.<br>• Setting SSEC0, SWC0 = 1, 0 is prohibited. | |

| SWC0 | Setting of the SNOOZE mode |
|---|---|
| 0 | Do not use the SNOOZE mode function. |
| 1 | Use the SNOOZE mode function. |
| • When there is a hardware trigger signal in the STOP mode, the STOP mode is exited, and A/D conversion is performed without operating the CPU (the SNOOZE mode).<br>• The SNOOZE mode function can only be specified when the high-speed on-chip oscillator clock is selected for the CPU/peripheral hardware clock ($f_{CLK}$). If any other clock is selected, specifying this mode is prohibited.<br>• Even when using SNOOZE mode, be sure to set the SWC0 bit to 0 in normal operation mode and change it to 1 just before shifting to STOP mode.<br>Also, be sure to change the SWC0 bit to 0 after returning from STOP mode to normal operation mode. | |

<R>

**Figure 12-19. Interrupt Using the UART Reception with SNOOZE Mode**

| EOC0n Bit | SSEC0 Bit | Normal Reception | Reception Error |
|---|---|---|---|
| 0 | 0 | INTSR0 is generated. | INTSR0 is generated. |
| 0 | 1 | INTSR0 is generated. | INTSR0 is generated. |
| 1 | 0 | INTSR0 is generated. | INTSRE0 is generated. |
| 1 | 1 | INTSR0 is generated. | No interrupt is generated. |

### 12.3.15 Input switch control register (ISC)

The ISC1 and ISC0 bits of the ISC register are used to realize a LIN-bus communication operation by UART2 in coordination with an external interrupt and the timer array unit.

When bit 0 is set to 1, the input signal of the serial data input (RxD2) pin is selected as an external interrupt (INTP0) that can be used to detect a wakeup signal.

When bit 1 is set to 1, the input signal of the serial data input (RxD2) pin is selected as a timer input, so that wake up signal can be detected, the low width of the break field, and the pulse width of the sync field can be measured by the timer.

The ISC register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears the ISC register to 00H.

**Figure 12-20. Format of Input Switch Control Register (ISC)**

Address: F0073H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ISC | 0 | 0 | 0 | 0 | 0 | 0 | ISC1 | ISC0 |

<R>

| ISC1 | Switching channel 7 input of timer array unit |
|---|---|
| 0 | 48- and 64-pin products:<br>Uses the input signal of the TI07 pin as a timer input (normal operation).<br>25- and 32-pin products:<br>Do not use a timer input signal for channel 7. |
| 1 | Input signal of the RxD2 pin is used as timer input (detects the wakeup signal and measures the low width of the break field and the pulse width of the sync field).<br>Setting is prohibited in the 25-pin products. |

| ISC0 | Switching external interrupt (INTP0) input |
|---|---|
| 0 | Uses the input signal of the INTP0 pin as an external interrupt (normal operation). |
| 1 | Uses the input signal of the RxD2 pin as an external interrupt (wakeup signal detection). |

**Caution Be sure to clear bits 7 to 2 to "0".**

### 12.3.16 Noise filter enable register 0 (NFEN0)

The NFEN0 register is used to set whether the noise filter can be used for the input signal from the serial data input pin to each channel.

Disable the noise filter of the pin used for CSI or simplified $I^2C$ communication, by clearing the corresponding bit of this register to 0.

Enable the noise filter of the pin used for UART communication, by setting the corresponding bit of this register to 1.

<R> When the noise filter is enabled, after synchronization is performed with the operation clock ($f_{MCK}$) of the target channel, 2-clock match detection is performed. When the noise filter is OFF, only synchronization is performed with the operation clock ($f_{MCK}$) of the target channel.

The NFEN0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears the NFEN0 register to 00H.

**Figure 12-21. Format of Noise Filter Enable Register 0 (NFEN0)**

Address: F0070H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| NFEN0 | 0 | 0 | 0 | SNFEN20 | 0 | SNFEN10 | 0 | SNFEN00 |

| SNFEN20 | Use of noise filter of RxD2 pin (RxD2/SDA20/SI20/P14) |
|---|---|
| 0 | Noise filter OFF |
| 1 | Noise filter ON |
| Set SNFEN20 to 1 to use the RxD2 pin.<br>Clear SNFEN20 to 0 to use the other than RxD2 pin. | |

| SNFEN10 | Use of noise filter of RxD1 pin (RXD1/ANI16/SI10/SDA10/P03) |
|---|---|
| 0 | Noise filter OFF |
| 1 | Noise filter ON |
| Set the SNFEN10 bit to 1 to use the RxD1 pin.<br>Clear the SNFEN10 bit to 0 to use the other than RxD1 pin. | |

| SNFEN00 | Use of noise filter of RxD0 pin (RXD0/TOOLRXD/SDA00/SI00/P11) |
|---|---|
| 0 | Noise filter OFF |
| 1 | Noise filter ON |
| Set the SNFEN00 bit to 1 to use the RxD0 pin.<br>Clear the SNFEN00 bit to 0 to use the other than RxD0 pin. | |

**Caution Be sure to clear bits 7 to 5, 3, and 1 to "0".**

### 12.3.17 Registers controlling port functions of serial input/output pins

Using the serial array unit requires setting of the registers that control the port functions multiplexed on the target channel (port mode register (PMxx), port register (Pxx), port input mode register (PIMxx), port output mode register (POMxx), port mode control register (PMCxx)).

For details, see **4.3.1 Port mode registers (PMxx)**, **4.3.2 Port registers (Pxx)**, **4.3.4 Port input mode registers (PIMxx)**, **4.3.5 Port output mode registers (POMxx)**, and **4.3.6 Port mode control registers (PMCxx)**.

Specifically, using a port pin with a multiplexed serial data or serial clock output function (e.g. P02/ANI17/SO10/TxD1) for serial data or serial clock output, requires setting the corresponding bits in the port mode control register (PMCxx) and port mode register (PMxx) to 0, and the corresponding bit in the port register (Pxx) to 1.

When using the port pin in N-ch open-drain output ($V_{DD}$ tolerance[Note 1]/$EV_{DD}$ tolerance[Note 2]) mode, set the corresponding bit in the port output mode register (POMxx) to 1. When connecting an external device operating on a different potential (1.8 V or 2.5 V), see **4.4.5 Handling different potential (1.8 V, 2.5 V) by using I/O buffers**.

Example: When P02/ANI17/SO10/TxD1 is to be used for serial data output
Set the PMC02 bit of port mode control register 0 to 0.
Set the PM02 bit of port mode register 0 to 0.
Set the P02 bit of port register 0 to 1.

Specifically, using a port pin with a multiplexed serial data or serial clock input function (e.g. P03/ANI16/SI10/RxD1/SDA10) for serial data or serial clock input, requires setting the corresponding bit in the port mode register (PMxx) to 1, and the corresponding bit in the port mode control register (PMCxx) to 0. In this case, the corresponding bit in the port register (Pxx) can be set to 0 or 1.

When the TTL input buffer is selected, set the corresponding bit in the port input mode register (PIMxx) to 1. When connecting an external device operating on a different potential (1.8 V or 2.5 V), see **4.4.5 Handling different potential (1.8 V, 2.5 V) by using I/O buffers**.

Example: When P03/ANI16/SI10/RxD1/SDA10 is to be used for serial data input
Set the PMC03 bit of port mode control register 0 to 0.
Set the PM03 bit of port mode register 0 to 1.
Set the P03 bit of port register 0 to 0 or 1.

**Notes 1.** 25 to 48-pin products
**2.** 64-pin product

## 12.4 Operation Stop Mode

Each serial interface of serial array unit has the operation stop mode.

In this mode, serial communication cannot be executed, thus reducing the power consumption.

In addition, the pin for serial interface can be used as port function pins in this mode.

### 12.4.1 Stopping the operation by units

The stopping of the operation by units is set by using peripheral enable register 0 (PER0).

The PER0 register is used to enable or disable supplying the clock to the peripheral hardware. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

To stop the operation of serial array unit 0, set bit 2 (SAU0EN) to 0.

To stop the operation of serial array unit 1, set bit 3 (SAU1EN) to 0.

**Figure 12-22. Peripheral Enable Register 0 (PER0) Setting When Stopping the Operation by Units**

**(a) Peripheral enable register 0 (PER0) … Set only the bit of SAUm to be stopped to 0.**

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PER0 | RTCEN | 0 | ADCEN | IICA0EN | SAU1EN[Note] | SAU0EN | 0 | TAU0EN |
| | × | × | × | × | 0/1 | 0/1 | × | × |

Control of SAUm input clock
0: Stops supply of input clock
1: Supplies input clock

**Note** 32, 48, and 64-pin products only.

**Cautions 1. If SAUmEN = 0, writing to a control register of serial array unit m is ignored, and, even if the register is read, only the default value is read**
**Note that this does not apply to the following registers.**
- **Input switch control register (ISC)**
- **Noise filter enable register 0 (NFEN0)**
- **Port input mode registers 0, 1 (PIM0, PIM1)**
- **Port output mode registers 0, 1, 5, 7 (POM0, POM1, POM5, POM7)**
- **Port mode control registers 0, 1, 3, 5, 7 (PMC0, PMC1, PMC3, PMC5, PMC7)**
- **Port mode registers 0, 1, 3, 5, 7 (PM0, PM1, PM3, PM5, PM7)**
- **Port registers 0, 1, 3, 5, 7 (P0, P1, P3, P5, P7)**

**2. Be sure to clear the following bits to 0.**
**25-pin products: bits 1, 3, 6**
**32, 48, 64-pin products: bits 1, 6**

**Remarks 1.** ×: Bits not used with serial array units (depending on the settings of other peripheral functions)
0/1: Set to 0 or 1 depending on the usage of the user

**2.** m: Unit number (m = 0, 1)

### 12.4.2 Stopping the operation by channels

The stopping of the operation by channels is set using each of the following registers.

**Figure 12-23. Each Register Setting When Stopping the Operation by Channels**

**(a) Serial channel stop register m (STm) … This register is a trigger register that is used to enable stopping communication/count by each channel.**

STm

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | STm3 Note 0/1 | STm2 Note 0/1 | STm1 0/1 | STm0 0/1 |

1: Clears the SEmn bit to 0 and stops the communication operation

* Because the STmn bit is a trigger bit, it is cleared immediately when SEmn = 0.

**(b) Serial Channel Enable Status Register m (SEm) … This register indicates whether data transmission/reception operation of each channel is enabled or stopped.**

SEm

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SEm3 Note 0/1 | SEm2 Note 0/1 | SEm1 0/1 | SEm0 0/1 |

0: Operation stops

* The SEm register is a read-only status register, whose operation is stopped by using the STm register. With a channel whose operation is stopped, the value of the CKOmn bit of the SOm register can be set by software.

**(c) Serial output enable register m (SOEm) … This register is a register that is used to enable or stop output of the serial communication operation of each channel.**

SOEm

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SOEm3 Note 0/1 | SOEm2 Note 0/1 | SOEm1 0/1 | SOEm0 0/1 |

0: Stops output by serial communication operation

* For channel n, whose serial output is stopped, the SOmn bit value of the SOm register can be set by software.

**(d) Serial output register m (SOm) …This register is a buffer register for serial output of each channel.**

SOm

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | CKOm3 Note 0/1 | CKOm2 Note 0/1 | CKOm1 0/1 | CKOm0 0/1 | 0 | 0 | 0 | 0 | SOm3 Note 0/1 | SOm2 Note 0/1 | SOm1 0/1 | SOm0 0/1 |

1: Serial clock output value is "1" 1: Serial data output value is "1"

* When using pins corresponding to each channel as port function pins, set the corresponding CKOmn, SOmn bits to "1".

**Note** Serial array unit 0 only.

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

**2.** ☐ : Setting disabled (fixed by hardware), 0/1: Set to 0 or 1 depending on the usage of the user

## 12.5 Operation of 3-Wire Serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) Communication

This is a clocked communication function that uses three lines: serial clock (SCK) and serial data (SI and SO) lines.

<R> [Data transmission/reception]

- Data length of 7 or 8 bits
- Phase control of transmit/receive data
- MSB/LSB first selectable

[Clock control]

- Master/slave selection
- Phase control of I/O clock
- Setting of transfer period by prescaler and internal counter of each channel

<R>
- Maximum transfer rate[Note]

  During master communication: Max. $f_{CLK}$/2 (CSI00 only)
  Max. $f_{CLK}$/4
  During slave communication: Max. $f_{MCK}$/6

[Interrupt function]

- Transfer end interrupt/buffer empty interrupt

[Error detection flag]

- Overrun error

In addition, CSI00 of following channels supports the SNOOZE mode. When SCK input is detected while in the STOP mode, the SNOOZE mode makes data reception that does not require the CPU possible.

<R> **Note** Use the clocks within a range satisfying the SCK cycle time ($t_{KCY}$) characteristics. For details, see **CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C)**, **CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to +105°C)**.

The channels supporting 3-wire serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) are channels 0 to 3 of SAU0 and channels 0 to 3 of SAU1.

- 25-pin products

| Unit | Channel | Used as CSI | Used as UART | Used as Simplified $I^2C$ |
|---|---|---|---|---|
| 0 | 0 | CSI00 | UART0 | IIC00 |
| | 1 | – | | – |
| | 2 | – | UART1 | – |
| | 3 | CSI11 | | IIC11 |

- 32-pin products

| Unit | Channel | Used as CSI | Used as UART | Used as Simplified $I^2C$ |
|---|---|---|---|---|
| 0 | 0 | CSI00 | UART0 | IIC00 |
| | 1 | – | | – |
| | 2 | – | UART1 | – |
| | 3 | CSI11 | | IIC11 |
| 1 | 0 | CSI20 | UART2 (supporting LIN-bus) | IIC20 |
| | 1 | – | | – |

- 48-pin products

| Unit | Channel | Used as CSI | Used as UART | Used as Simplified $I^2C$ |
|---|---|---|---|---|
| 0 | 0 | CSI00 | UART0 | IIC00 |
| | 1 | CSI01 | | IIC01 |
| | 2 | – | UART1 | – |
| | 3 | CSI11 | | IIC11 |
| 1 | 0 | CSI20 | UART2 (supporting LIN-bus) | IIC20 |
| | 1 | CSI21 | | IIC21 |

- 64-pin products

| Unit | Channel | Used as CSI | Used as UART | Used as Simplified $I^2C$ |
|---|---|---|---|---|
| 0 | 0 | CSI00 | UART0 | IIC00 |
| | 1 | CSI01 | | IIC01 |
| | 2 | CSI10 | UART1 | IIC10 |
| | 3 | CSI11 | | IIC11 |
| 1 | 0 | CSI20 | UART2 (supporting LIN-bus) | IIC20 |
| | 1 | CSI21 | | IIC21 |

3-wire serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) performs the following seven types of communication operations.

- Master transmission (See **12.5.1**.)
- Master reception (See **12.5.2**.)
- Master transmission/reception (See **12.5.3**.)
- Slave transmission (See **12.5.4**.)
- Slave reception (See **12.5.5**.)
- Slave transmission/reception (See **12.5.6**.)
- SNOOZE mode function (See **12.5.7**.)

### 12.5.1 Master transmission

<R> Master transmission is that the RL78 microcontroller outputs a transfer clock and transmits data to another device.

| 3-Wire Serial I/O | CSI00 | CSI01 | CSI10 | CSI11 | CSI20 | CSI21 |
|---|---|---|---|---|---|---|
| Target channel | Channel 0 of SAU0 | Channel 1 of SAU0 | Channel 2 of SAU0 | Channel 3 of SAU0 | Channel 0 of SAU1 | Channel 1 of SAU1 |
| Pins used | SCK00, SO00 | SCK01, SO01 | SCK10, SO10 | SCK11, SO11 | SCK20, SO20 | SCK21, SO21 |
| Interrupt | INTCSI00 | INTCSI01 | INTCSI10 | INTCSI11 | INTCSI20 | INTCSI21 |
| | Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected. | | | | | |
| Error detection flag | None | | | | | |
| Transfer data length | 7 or 8 bits | | | | | |
| <R> Transfer rate[Note] | Max. $f_{CLK}/2$ [MHz] (CSI00 only), $f_{CLK}/4$ [MHz]<br>Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [Hz][Note] $f_{CLK}$: System clock frequency | | | | | |
| Data phase | Selectable by the DAPmn bit of the SCRmn register<br>• DAPmn = 0: Data output starts from the start of the operation of the serial clock.<br>• DAPmn = 1: Data output starts half a clock before the start of the serial clock operation. | | | | | |
| Clock phase | Selectable by the CKPmn bit of the SCRmn register<br>• CKPmn = 0: Non-reverse<br>• CKPmn = 1: Reverse | | | | | |
| Data direction | MSB or LSB first | | | | | |

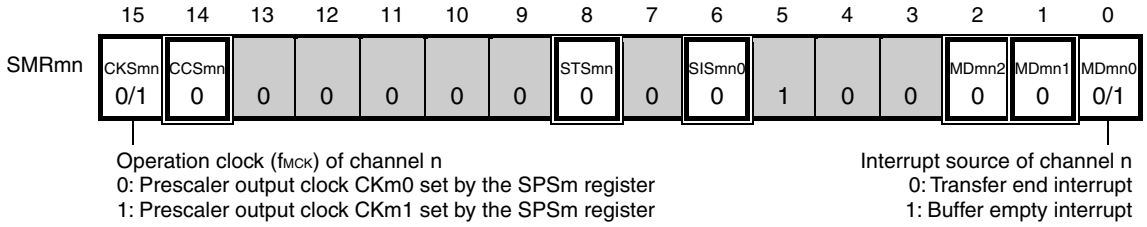
<R> **Note** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristic in the electrical specifications (see **CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C), CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to +105°C)**).

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

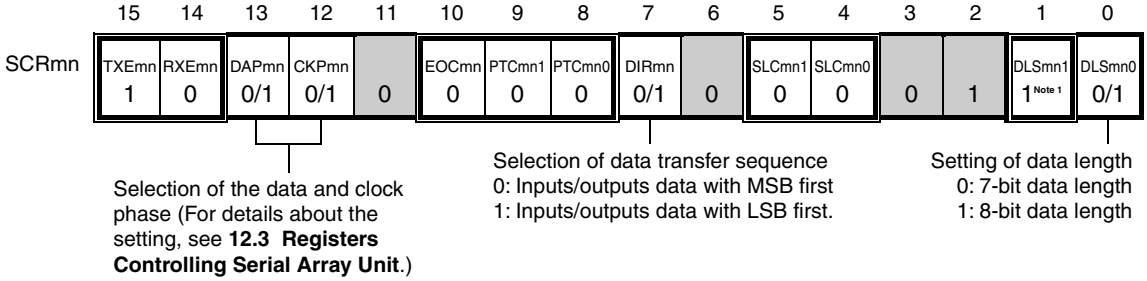
### (1) Register setting

**Figure 12-24. Example of Contents of Registers for Master Transmission of 3-Wire Serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) (1/2)**

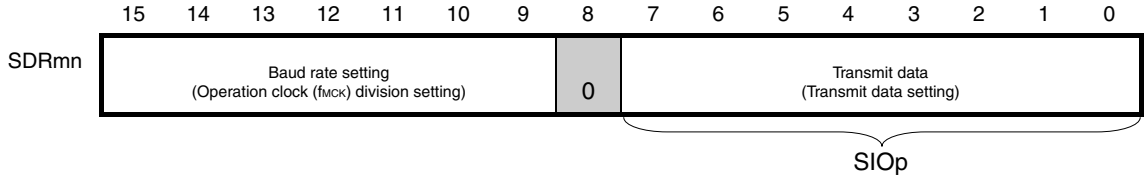
**(a) Serial mode register mn (SMRmn)**



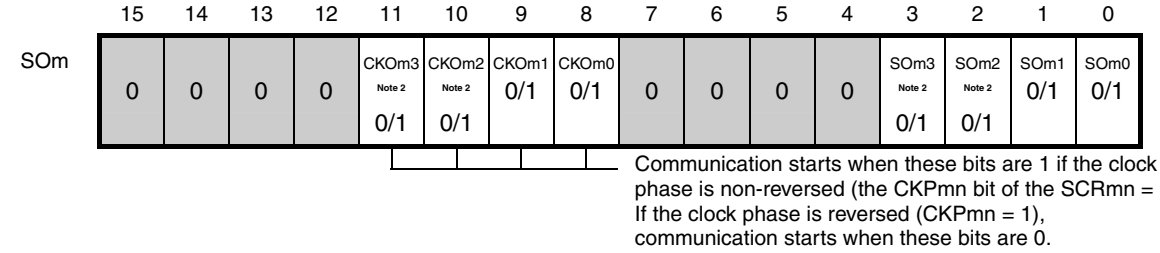

**(b) Serial communication operation setting register mn (SCRmn)**




**(c) Serial data register mn (SDRmn) (lower 8 bits: SIOp)**




**(d) Serial output register m (SOm) … Sets only the bits of the target channel.**




**Notes 1.** Only provided for the SCR00 and SCR01 registers. This bit is fixed to 1 for the other registers.

**2.** Unit 0 only

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3),
p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**2.** ☐: Setting is fixed in the CSI master transmission mode, ▒: Setting disabled (set to the initial value)
0/1: Set to 0 or 1 depending on the usage of the user

**Figure 12-24. Example of Contents of Registers for Master Transmission of 3-Wire Serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) (2/2)**

**(e) Serial output enable register m (SOEm) … Sets only the bits of the target channel to 1.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SOEm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SOEm3 Note 0/1 | SOEm2 Note 0/1 | SOEm1 0/1 | SOEm0 0/1 |

**(f) Serial channel start register m (SSm) … Sets only the bits of the target channel to 1.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SSm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SSm3 Note 0/1 | SSm2 Note 0/1 | SSm1 0/1 | SSm0 0/1 |

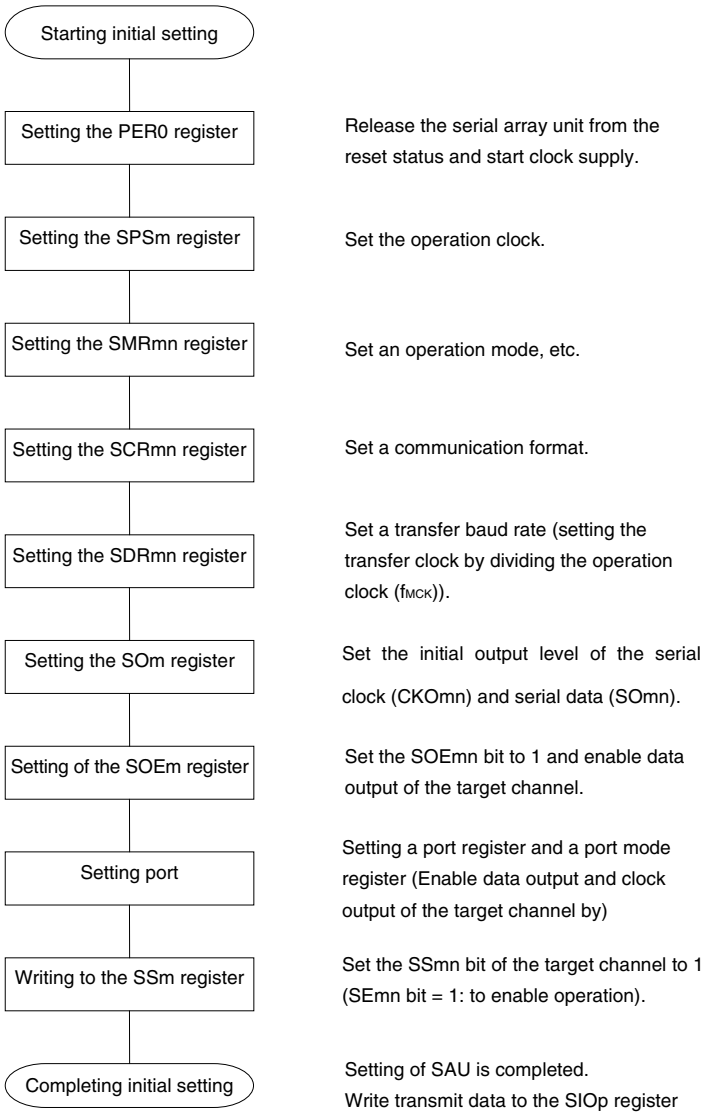
**Note** Unit 0 only

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**2.** ☐: Setting disabled (set to the initial value)
×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)
0/1: Set to 0 or 1 depending on the usage of the user

### (2) Operation procedure

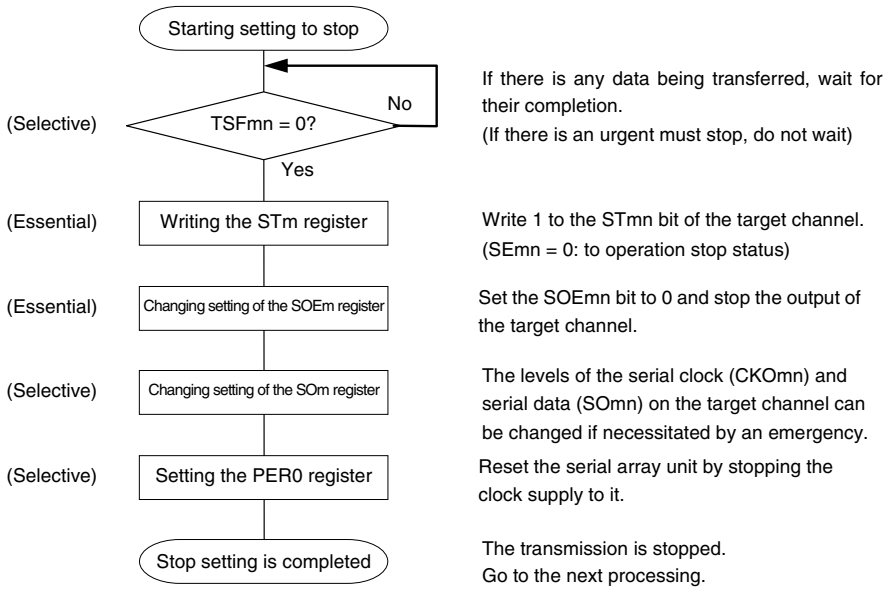
**Figure 12-25. Initial Setting Procedure for Master Transmission**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

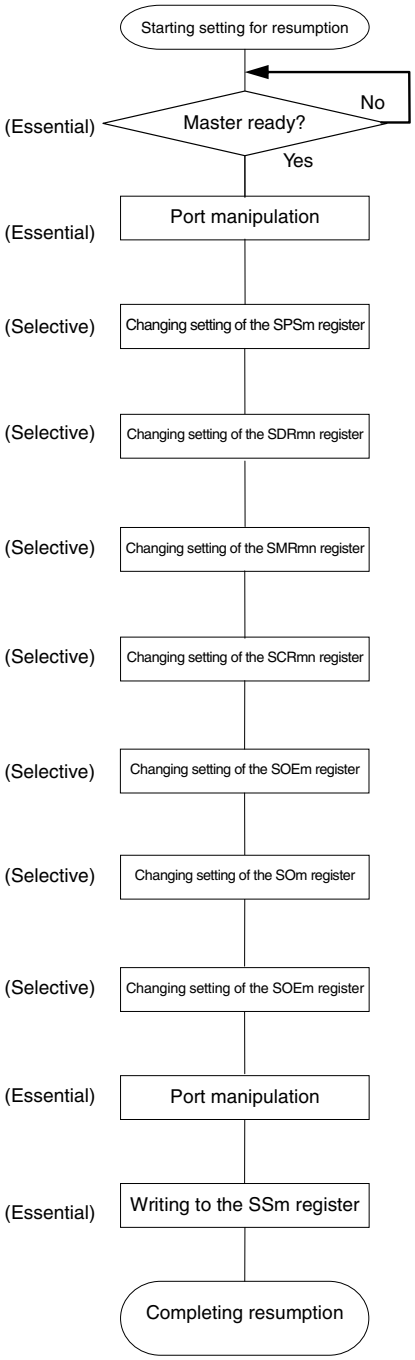
**Figure 12-26. Procedure for Stopping Master Transmission**

<R>




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**Figure 12-27. Procedure for Resuming Master Transmission**




Wait until stop the communication target (slave) or communication operation completed

Disable data output and clock output of the target channel by setting a port register and a port mode register.

Re-set the register to change the operation clock setting.

Re-set the register to change the transfer baud rate setting (setting the transfer clock by dividing the operation clock ($f_{MCK}$)).

Re-set the register to change serial mode register mn (SMRmn) setting.

Re-set the register to change serial communication operation setting register mn (SCRmn) setting.

Set the SOEmn bit to 0 to stop output from the target channel.

Set the initial output level of the serial clock (CKOmn) and serial data (SOmn).

Set the SOEmn bit to 1 and enable output from the target channel.

Enable data output and clock output of the target channel by setting a port register and a port mode register.

Set the SSmn bit of the target channel to 1 (SEmn = 1: to enable operation).

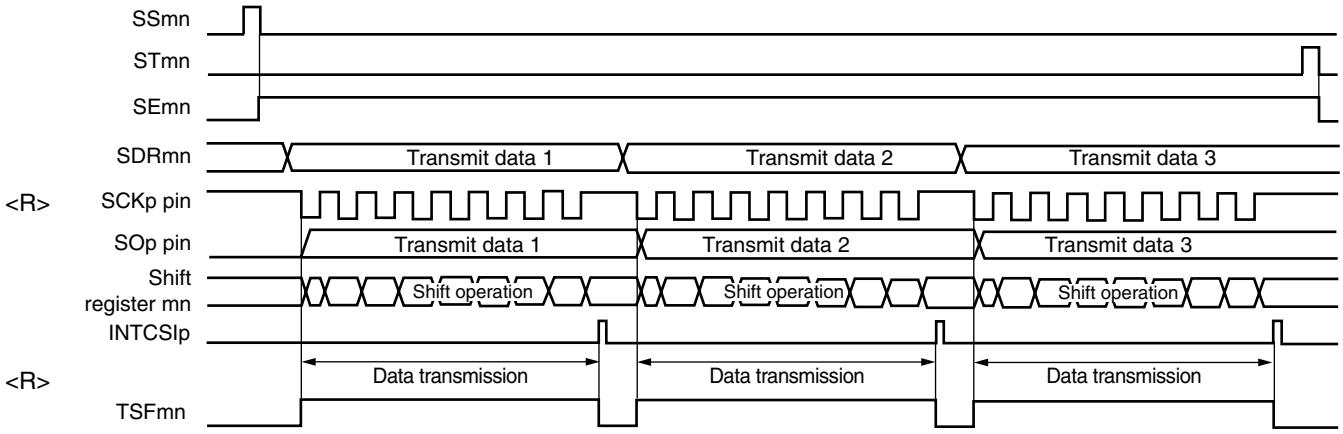
Setting is completed
Sets transmit data to the SIOp register (bits 7 to 0 of the SDRmn register) and start communication.

**Remarks 1.** If PER0 is rewritten while stopping the master transmission and the clock supply is stopped, wait until the transmission target (slave) stops or transmission finishes, and then perform initialization instead of restarting the transmission.

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

### (3) Processing flow (in single-transmission mode)

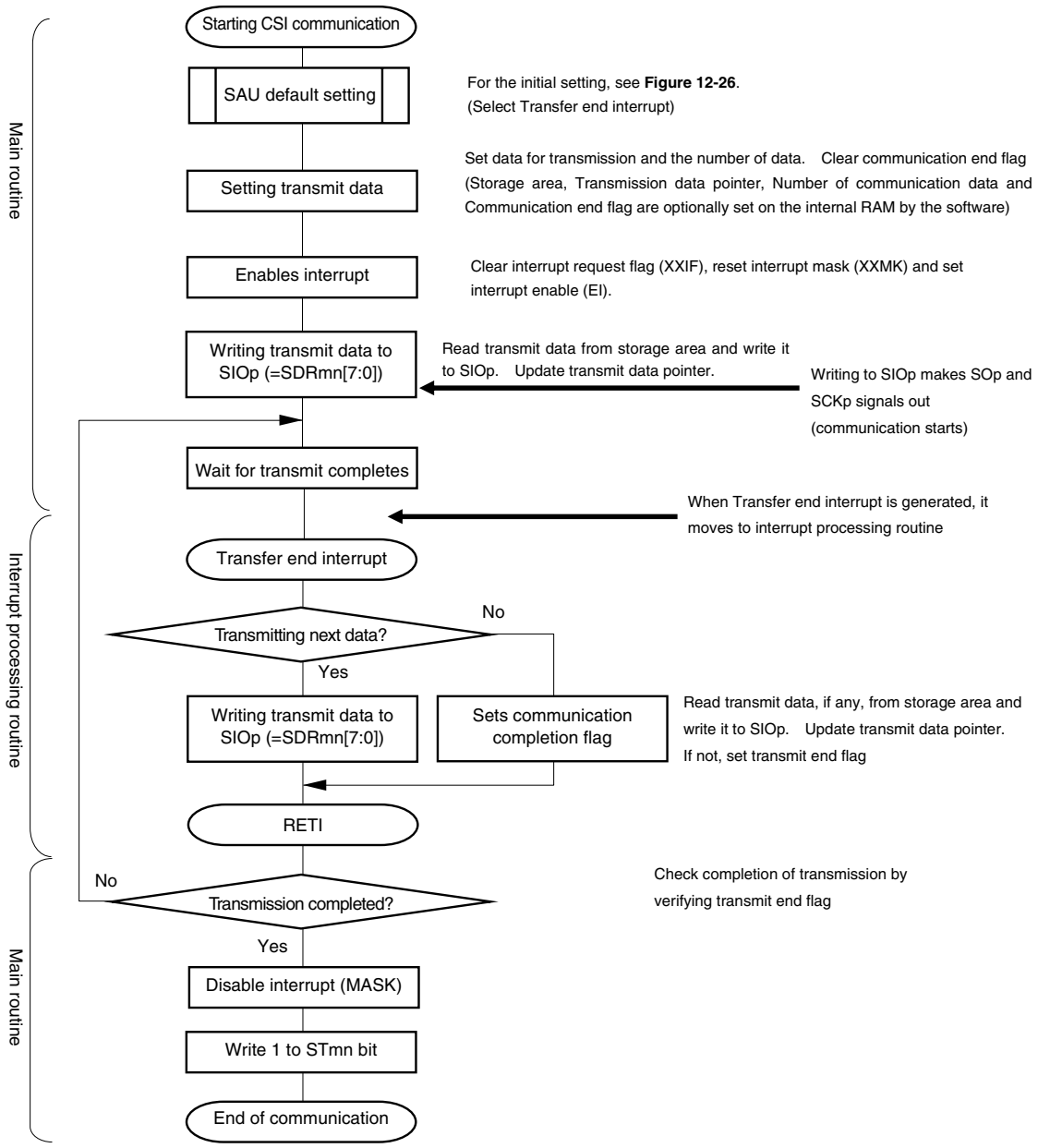
**Figure 12-28. Timing Chart of Master Transmission (in Single-Transmission Mode)**
**(Type 1: DAPmn = 0, CKPmn = 0)**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**Figure 12-29. Flowchart of Master Transmission (in Single-Transmission Mode)**

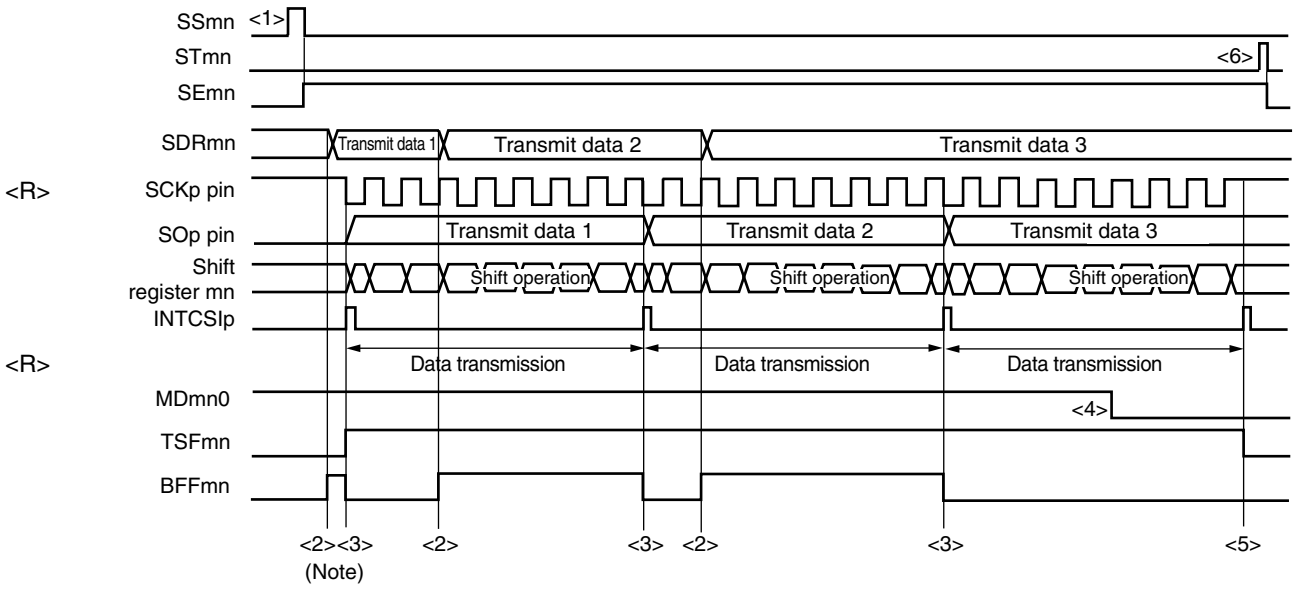
<R>




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

### (4) Processing flow (in continuous transmission mode)

**Figure 12-30. Timing Chart of Master Transmission (in Continuous Transmission Mode)**
**(Type 1: DAPmn = 0, CKPmn = 0)**

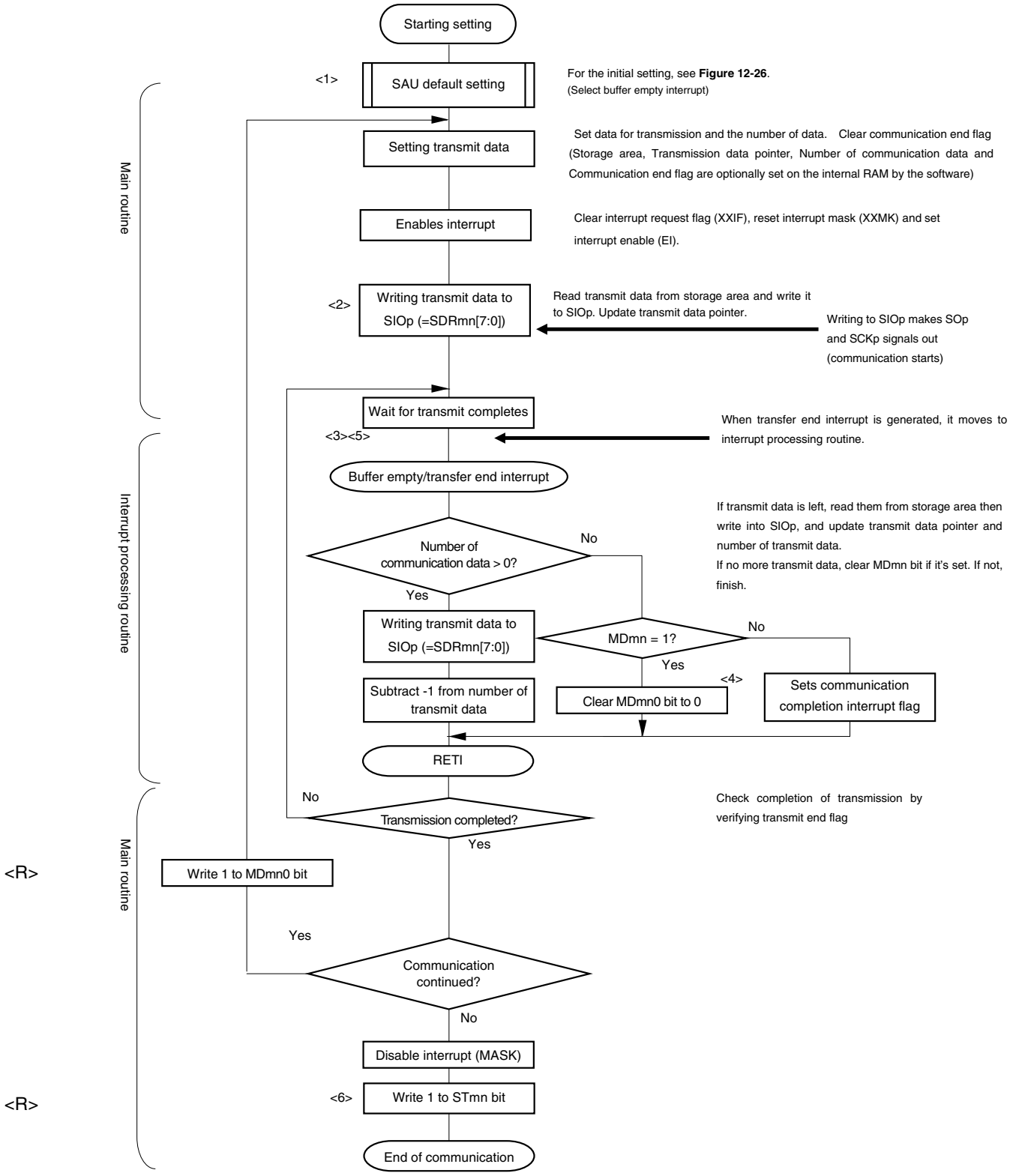



**Note** If transmit data is written to the SDRmn register while the BFFmn bit of serial status register mn (SSRmn) is 1 (valid data is stored in serial data register mn (SDRmn)), the transmit data is overwritten.

**Caution The MDmn0 bit of serial mode register mn (SMRmn) can be rewritten even during operation. However, rewrite it before transfer of the last bit is started, so that it will be rewritten before the transfer end interrupt of the last transmit data.**

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**Figure 12-31. Flowchart of Master Transmission (in Continuous Transmission Mode)**




**Remarks 1.** <1> to <6> in the figure correspond to <1> to <6> in **Figure 12-30 Timing Chart of Master Transmission (in Continuous Transmission Mode)**.

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

### 12.5.2 Master reception

<R> Master reception is that the RL78 microcontroller outputs a transfer clock and receives data from other device.

| 3-Wire Serial I/O | CSI00 | CSI01 | CSI10 | CSI11 | CSI20 | CSI21 |
|---|---|---|---|---|---|---|
| Target channel | Channel 0 of SAU0 | Channel 1 of SAU0 | Channel 2 of SAU0 | Channel 3 of SAU0 | Channel 0 of SAU1 | Channel 1 of SAU1 |
| Pins used | SCK00, SI00 | SCK01, SI01 | SCK10, SI10 | SCK11, SI11 | SCK20, SI20 | SCK21, SI21 |
| Interrupt | INTCSI00 | INTCSI01 | INTCSI10 | INTCSI11 | INTCSI20 | INTCSI21 |
| | Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected. | | | | | |
| Error detection flag | Overrun error detection flag (OVFmn) only | | | | | |
| Transfer data length | 7 or 8 bits | | | | | |
| <R> Transfer rate[Note] | Max. $f_{CLK}/2$ [MHz] (CSI00 only), $f_{CLK}/4$ [MHz]<br>Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [Hz][Note] $f_{CLK}$: System clock frequency | | | | | |
| Data phase | Selectable by the DAPmn bit of the SCRmn register<br>• DAPmn = 0: Data input starts from the start of the operation of the serial clock.<br>• DAPmn = 1: Data input starts half a clock before the start of the serial clock operation. | | | | | |
| Clock phase | Selectable by the CKPmn bit of the SCRmn register<br>• CKPmn = 0: Non-reverse<br>• CKPmn = 1: Reverse | | | | | |
| Data direction | MSB or LSB first | | | | | |

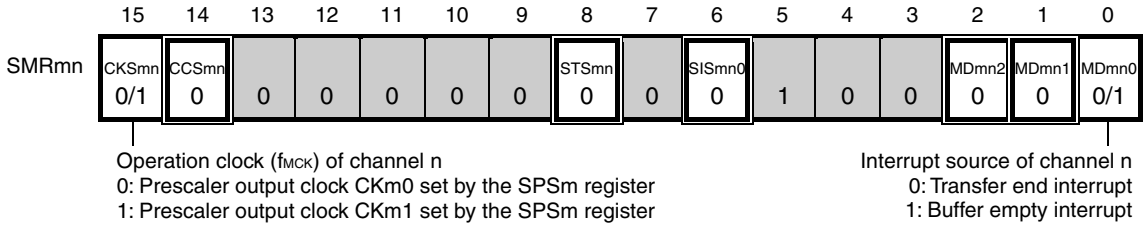
**Note** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristic in the electrical specifications (see **CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C), CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to +105°C)**).

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

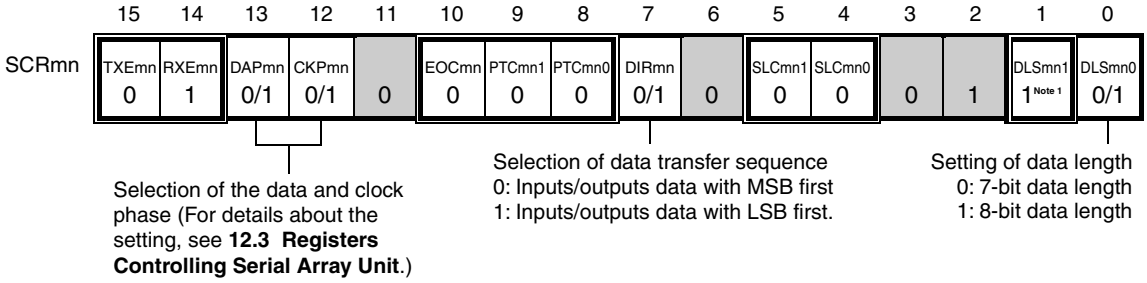
### (1) Register setting

**Figure 12-32. Example of Contents of Registers for Master Reception of 3-Wire Serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) (1/2)**

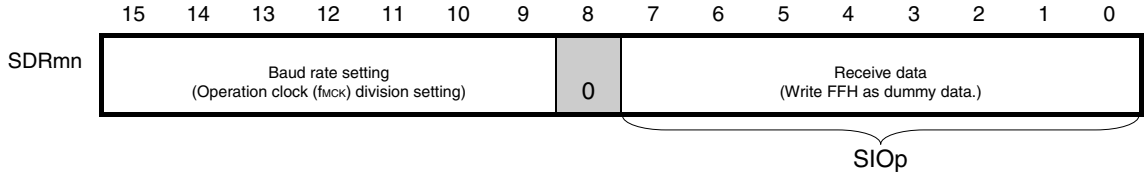
**(a) Serial mode register mn (SMRmn)**



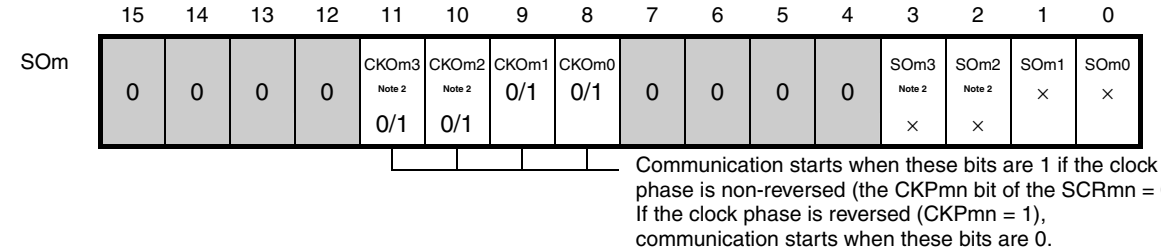

**(b) Serial communication operation setting register mn (SCRmn)**




**(c) Serial data register mn (SDRmn) (lower 8 bits: SIOp)**




**(d) Serial output register m (SOm) … Sets only the bits of the target channel.**




**Notes 1.** Only provided for the SCR00 and SCR01 registers. This bit is fixed to 1 for the other registers.

**2.** Unit 0 only

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**2.** ☐ : Setting is fixed in the CSI master reception mode, ▒ : Setting disabled (set to the initial value)
×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)
0/1: Set to 0 or 1 depending on the usage of the user

**Figure 12-32. Example of Contents of Registers for Master Reception of 3-Wire Serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21, CSI30, CSI31) (2/2)**

**(e) Serial output enable register m (SOEm) …The register that not used in this mode.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SOEm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SOEm3 Note × | SOEm2 Note × | SOEm1 × | SOEm0 × |

**(f) Serial channel start register m (SSm) … Sets only the bits of the target channel to 1.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SSm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SSm3 Note 0/1 | SSm2 Note 0/1 | SSm1 0/1 | SSm0 0/1 |

**Note** Unit 0 only

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

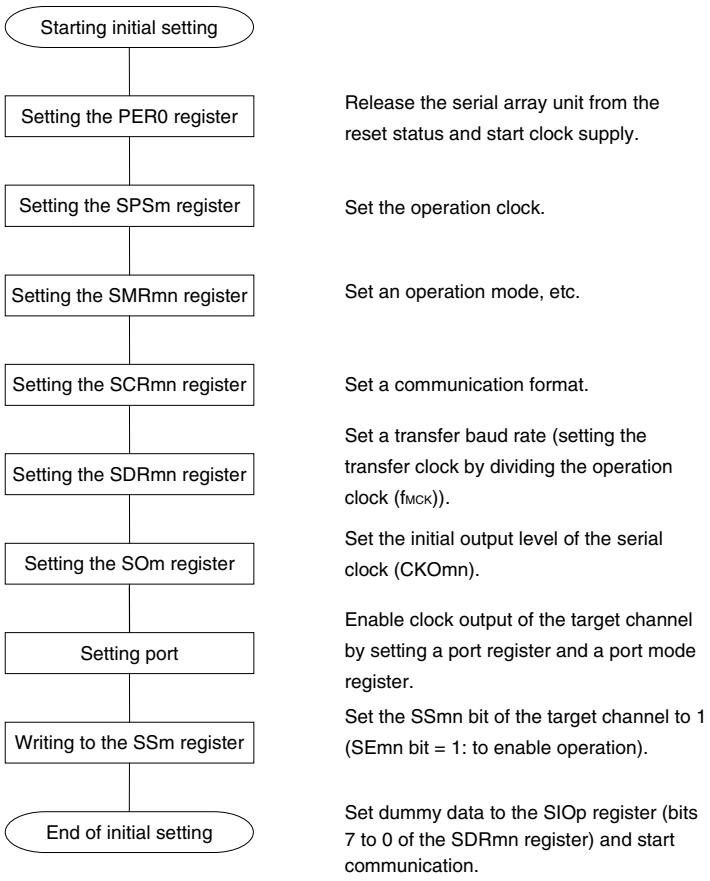
**2.** ☐ : Setting disabled (set to the initial value)

×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)

0/1: Set to 0 or 1 depending on the usage of the user

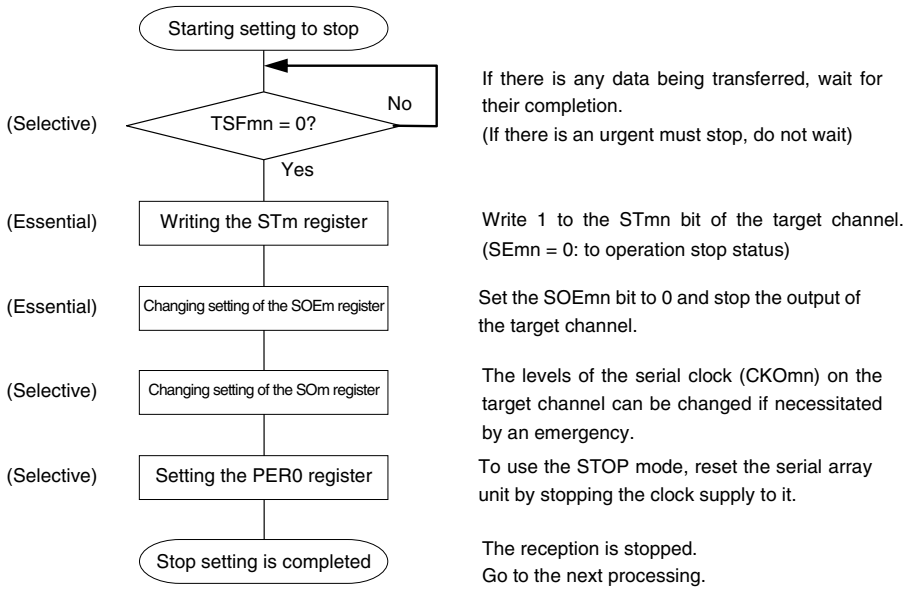
### (2) Operation procedure

**Figure 12-33. Initial Setting Procedure for Master Reception**



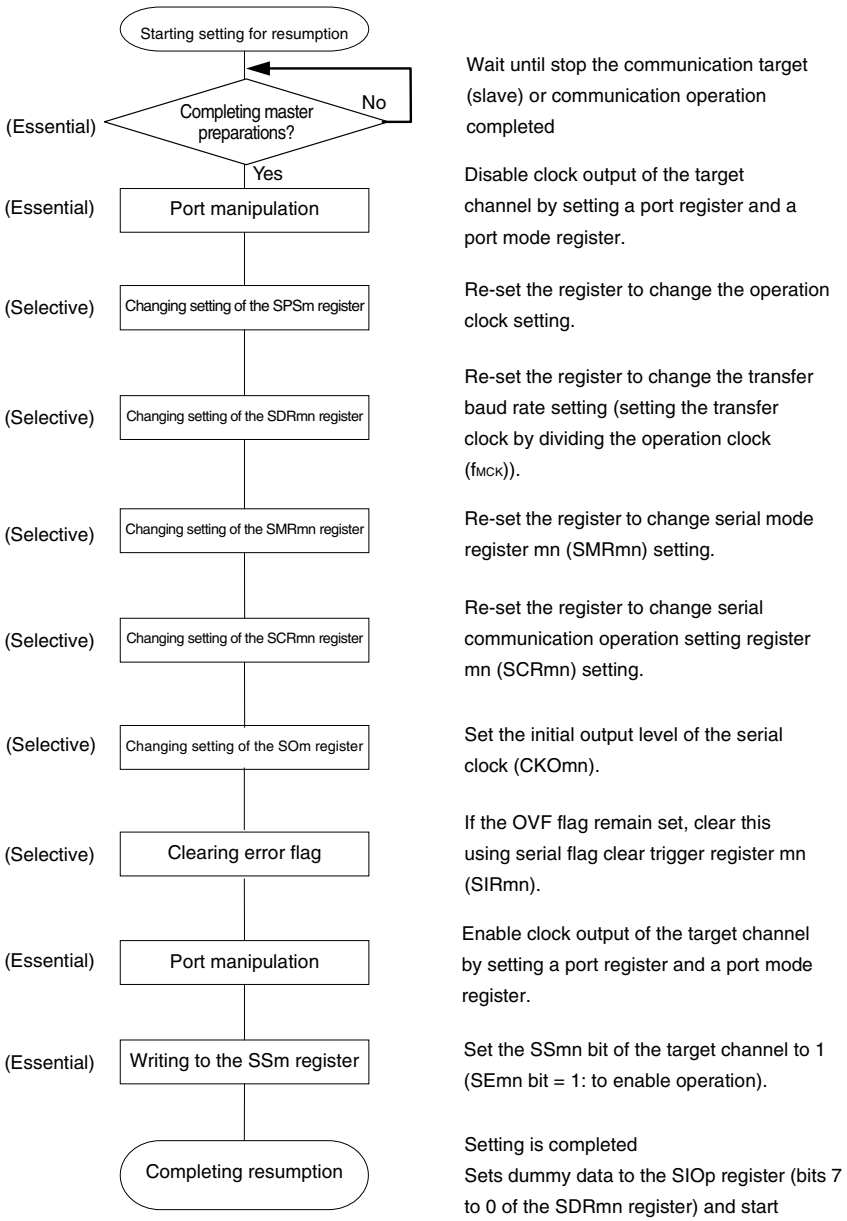

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**Figure 12-34. Procedure for Stopping Master Reception**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**Figure 12-35. Procedure for Resuming Master Reception**

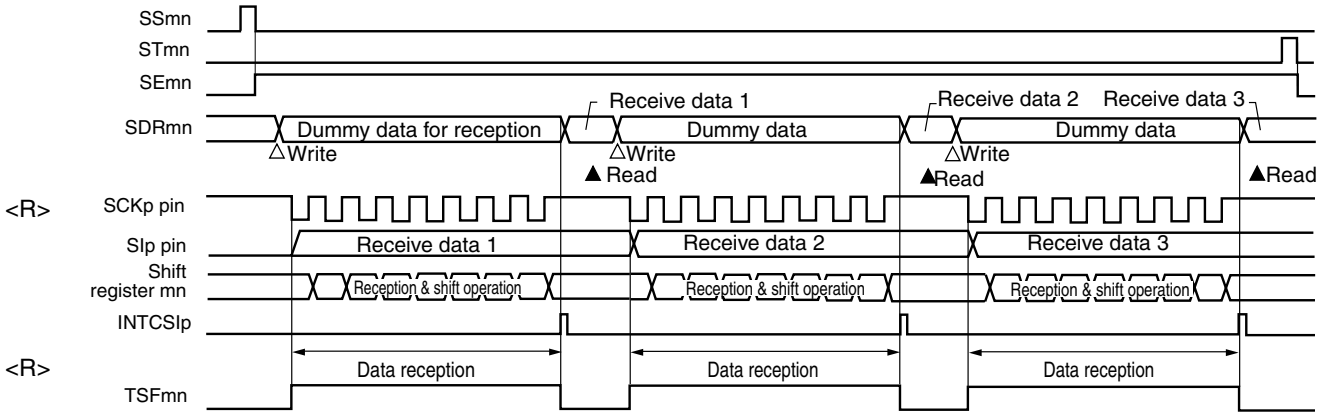



**Remarks 1.** If PER0 is rewritten while stopping the master transmission and the clock supply is stopped, wait until the transmission target (slave) stops or transmission finishes, and then perform initialization instead of restarting the transmission.

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3),
p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

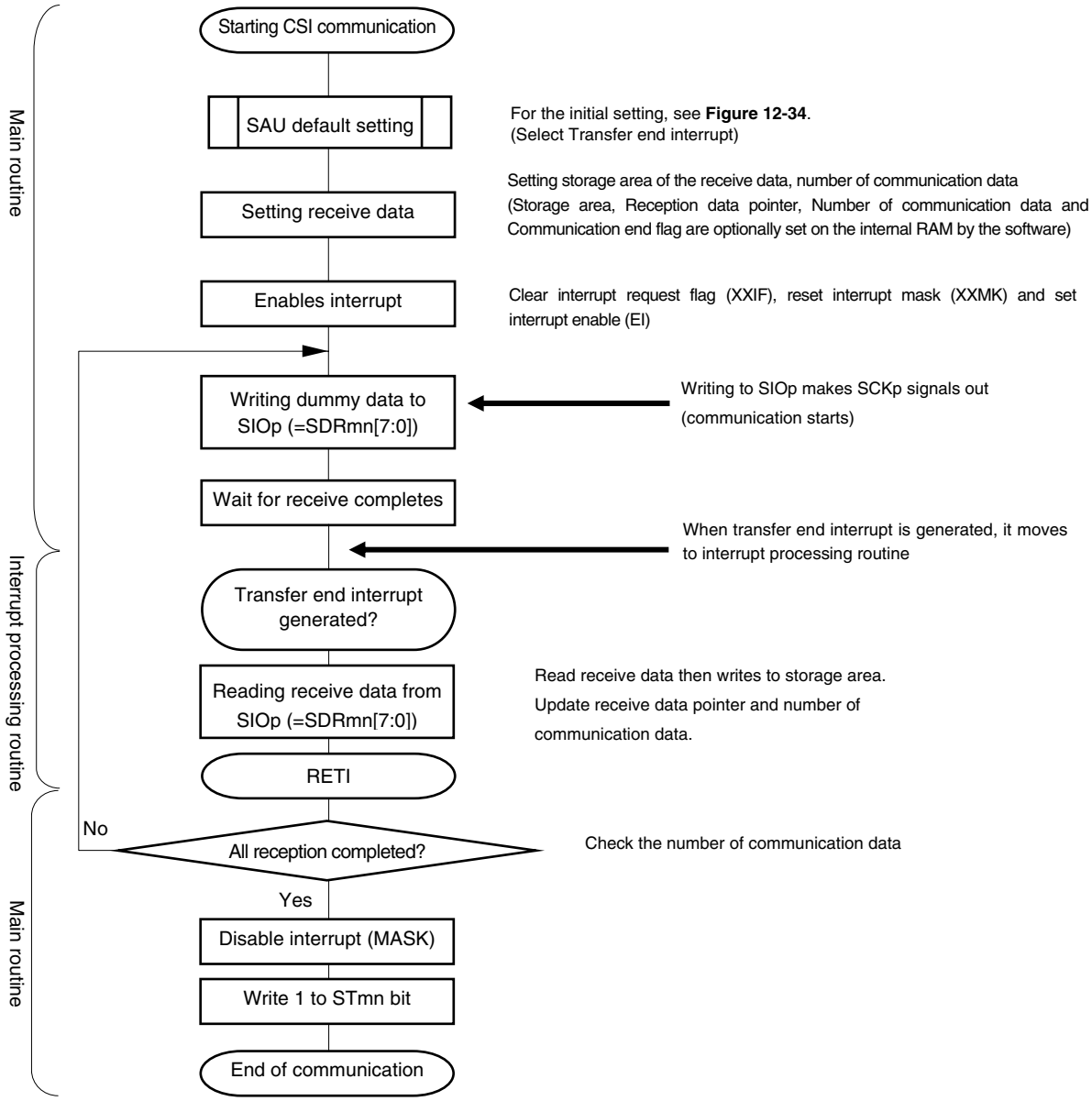
### (3) Processing flow (in single-reception mode)

**Figure 12-36. Timing Chart of Master Reception (in Single-Reception Mode)**
**(Type 1: DAPmn = 0, CKPmn = 0)**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**Figure 12-37. Flowchart of Master Reception (in Single-Reception Mode)**

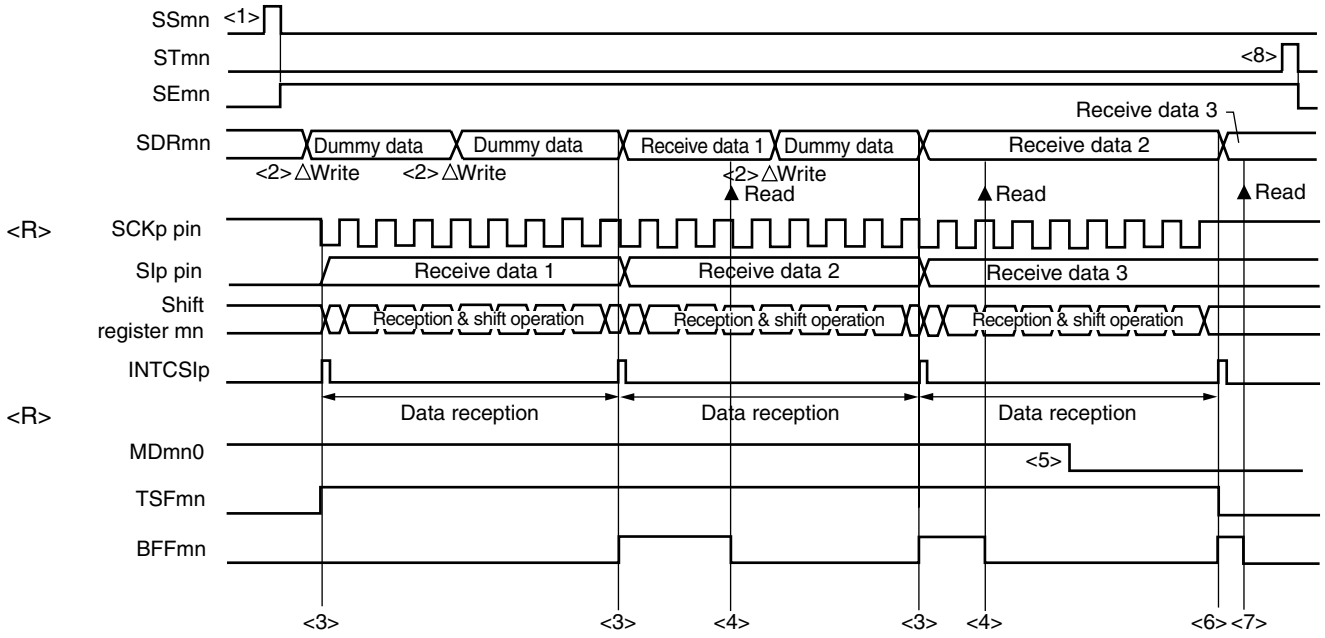



<R>

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

### (4) Processing flow (in continuous reception mode)

**Figure 12-38. Timing Chart of Master Reception (in Continuous Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)**

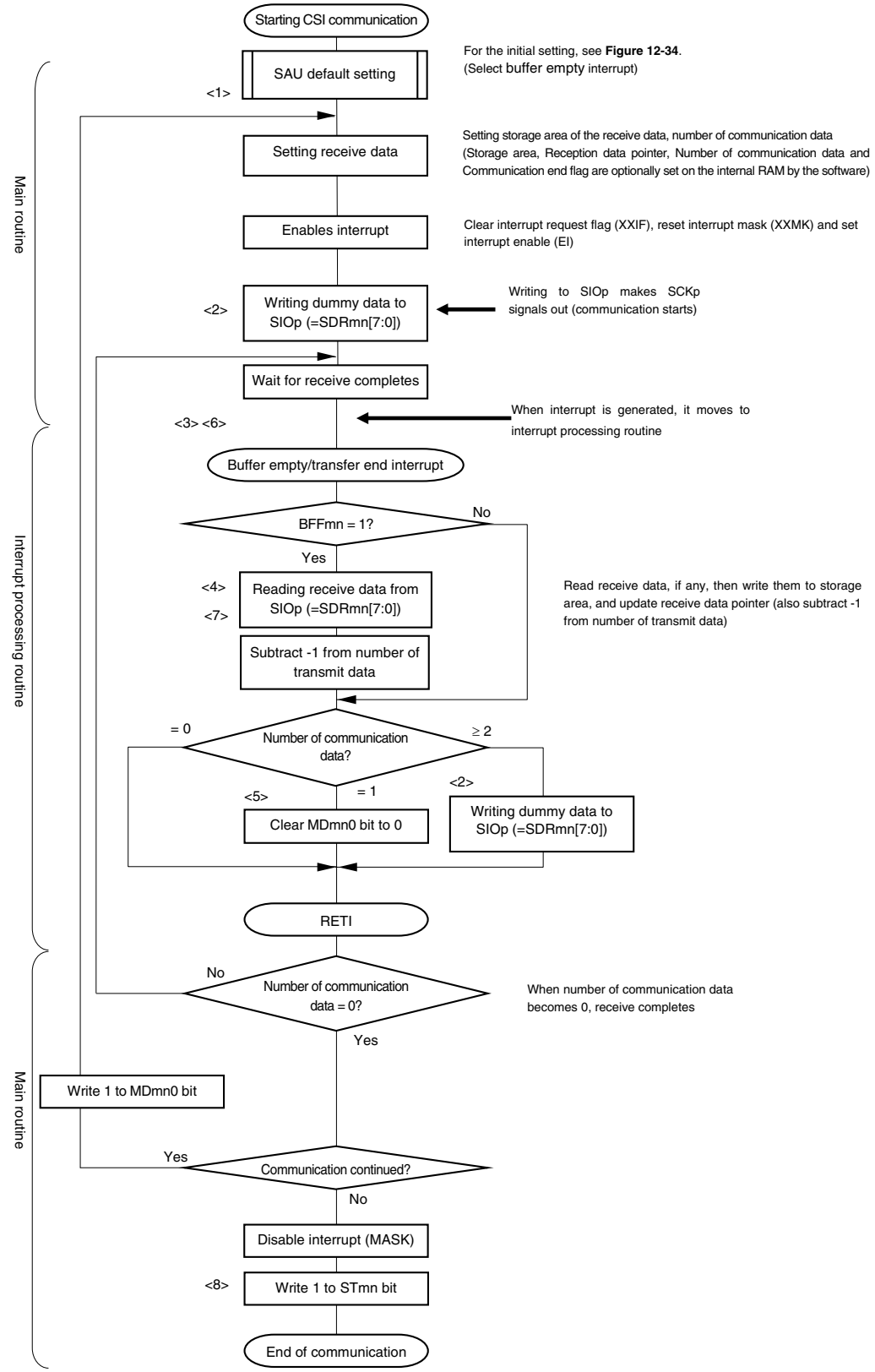



**Caution The MDmn0 bit can be rewritten even during operation.**
**However, rewrite it before receive of the last bit is started, so that it has been rewritten before the transfer end interrupt of the last receive data.**

**Remarks 1.** <1> to <8> in the figure correspond to <1> to <8> in **Figure 12-39 Flowchart of Master Reception (in Continuous Reception Mode)**.

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**Figure 12-39. Flowchart of Master Reception (in Continuous Reception Mode)**




<R>

<R>

**Remarks 1.** <1> to <8> in the figure correspond to <1> to <8> in **Figure 12-38 Timing Chart of Master Reception (in Continuous Reception Mode)**.

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

### 12.5.3 Master transmission/reception

<R> Master transmission/reception is that the RL78 microcontroller outputs a transfer clock and transmits/receives data to/from other device.

| 3-Wire Serial I/O | CSI00 | CSI01 | CSI10 | CSI11 | CSI20 | CSI21 |
|---|---|---|---|---|---|---|
| Target channel | Channel 0 of SAU0 | Channel 1 of SAU0 | Channel 2 of SAU0 | Channel 3 of SAU0 | Channel 0 of SAU1 | Channel 1 of SAU1 |
| Pins used | SCK00, SI00, SO00 | SCK01, SI01, SO01 | SCK10, SI10, SO10 | SCK11, SI11, SO11 | SCK20, SI20, SO20 | SCK21, SI21, SO21 |
| Interrupt | INTCSI00 | INTCSI01 | INTCSI10 | INTCSI11 | INTCSI20 | INTCSI21 |
| | Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected. | | | | | |
| Error detection flag | Overrun error detection flag (OVFmn) only | | | | | |
| Transfer data length | 7 or 8 bits | | | | | |
| <R> Transfer rate[Note] | Max. $f_{CLK}/2$ [MHz] (CSI00 only), $f_{CLK}/4$ [MHz]<br>Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [Hz][Note] $f_{CLK}$: System clock frequency | | | | | |
| Data phase | Selectable by the DAPmn bit of the SCRmn register<br>• DAPmn = 0: Data I/O starts at the start of the operation of the serial clock.<br>• DAPmn = 1: Data I/O starts half a clock before the start of the serial clock operation. | | | | | |
| Clock phase | Selectable by the CKPmn bit of the SCRmn register<br>• CKPmn = 0: Non-reverse<br>• CKPmn = 1: Reverse | | | | | |
| Data direction | MSB or LSB first | | | | | |

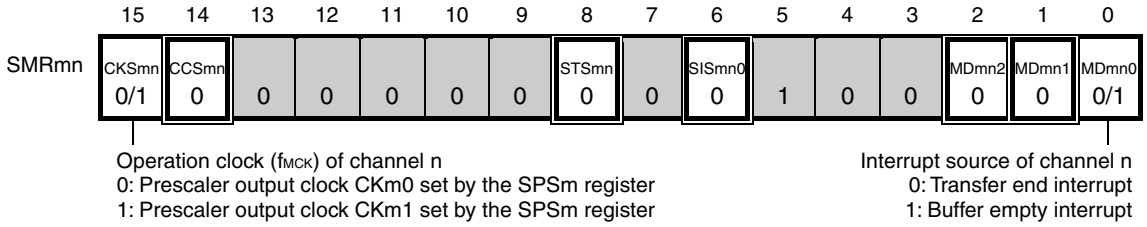
**Note** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristic in the electrical specifications (see **CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C), CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to +105°C)**).

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

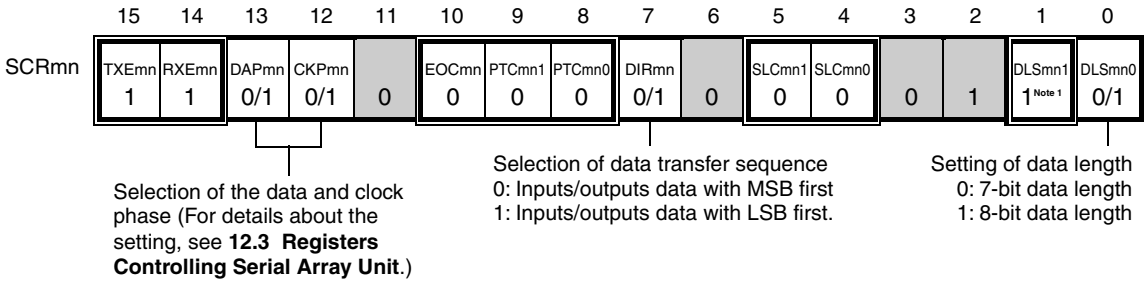
### (1) Register setting

**Figure 12-40. Example of Contents of Registers for Master Transmission/Reception of 3-Wire Serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21, CSI30, CSI31) (1/2)**

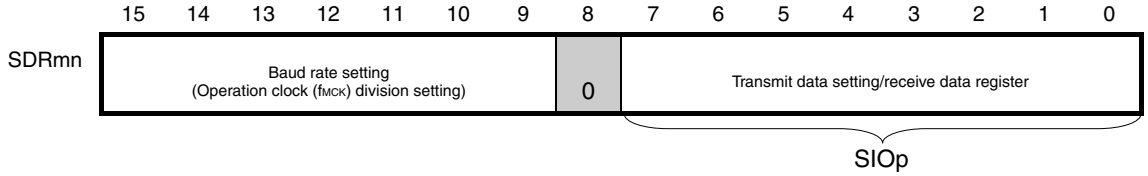
**(a) Serial mode register mn (SMRmn)**



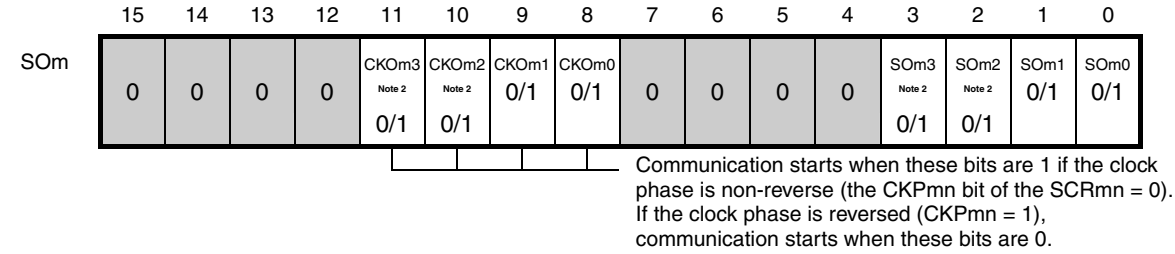

**(b) Serial communication operation setting register mn (SCRmn)**




**(c) Serial data register mn (SDRmn) (lower 8 bits: SIOp)**




**(d) Serial output register m (SOm) … Sets only the bits of the target channel.**




**Notes 1.** Only provided for the SCR00 and SCR01 registers. This bit is fixed to 1 for the other registers.

**2.** Unit 0 only

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**2.** ☐: Setting is fixed in the CSI master transmission/reception mode

: Setting disabled (set to the initial value)

0/1: Set to 0 or 1 depending on the usage of the user

**Figure 12-40. Example of Contents of Registers for Master Transmission/Reception of 3-Wire Serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21, CSI30, CSI31) (2/2)**

**(e) Serial output enable register m (SOEm) … Sets only the bits of the target channel to 1.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SOEm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SOEm3 Note 0/1 | SOEm2 Note 0/1 | SOEm1 0/1 | SOEm0 0/1 |

**(f) Serial channel start register m (SSm) … Sets only the bits of the target channel to 1.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SSm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SSm3 Note 0/1 | SSm2 Note 0/1 | SSm1 0/1 | SSm0 0/1 |

**Note** Serial array unit 0 only.

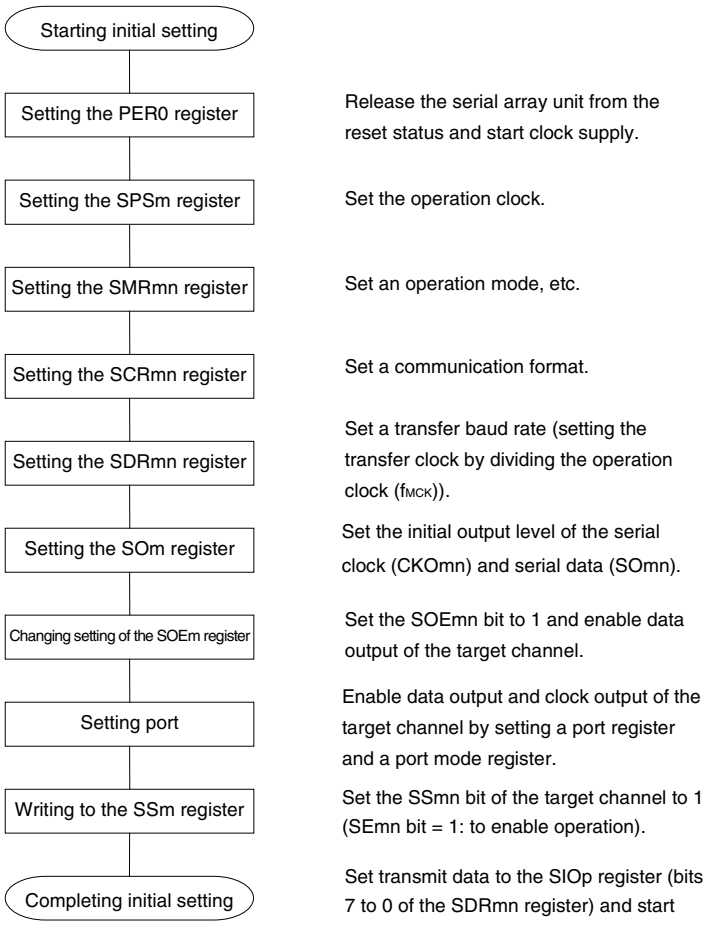
**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

**2.** ☐ : Setting disabled (set to the initial value)

0/1: Set to 0 or 1 depending on the usage of the user

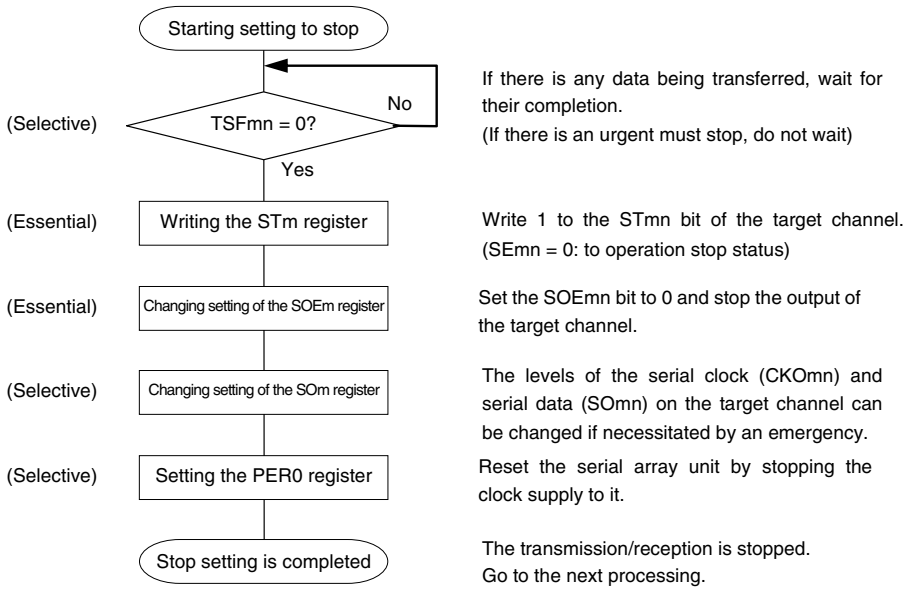
### (2) Operation procedure

**Figure 12-41. Initial Setting Procedure for Master Transmission/Reception**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

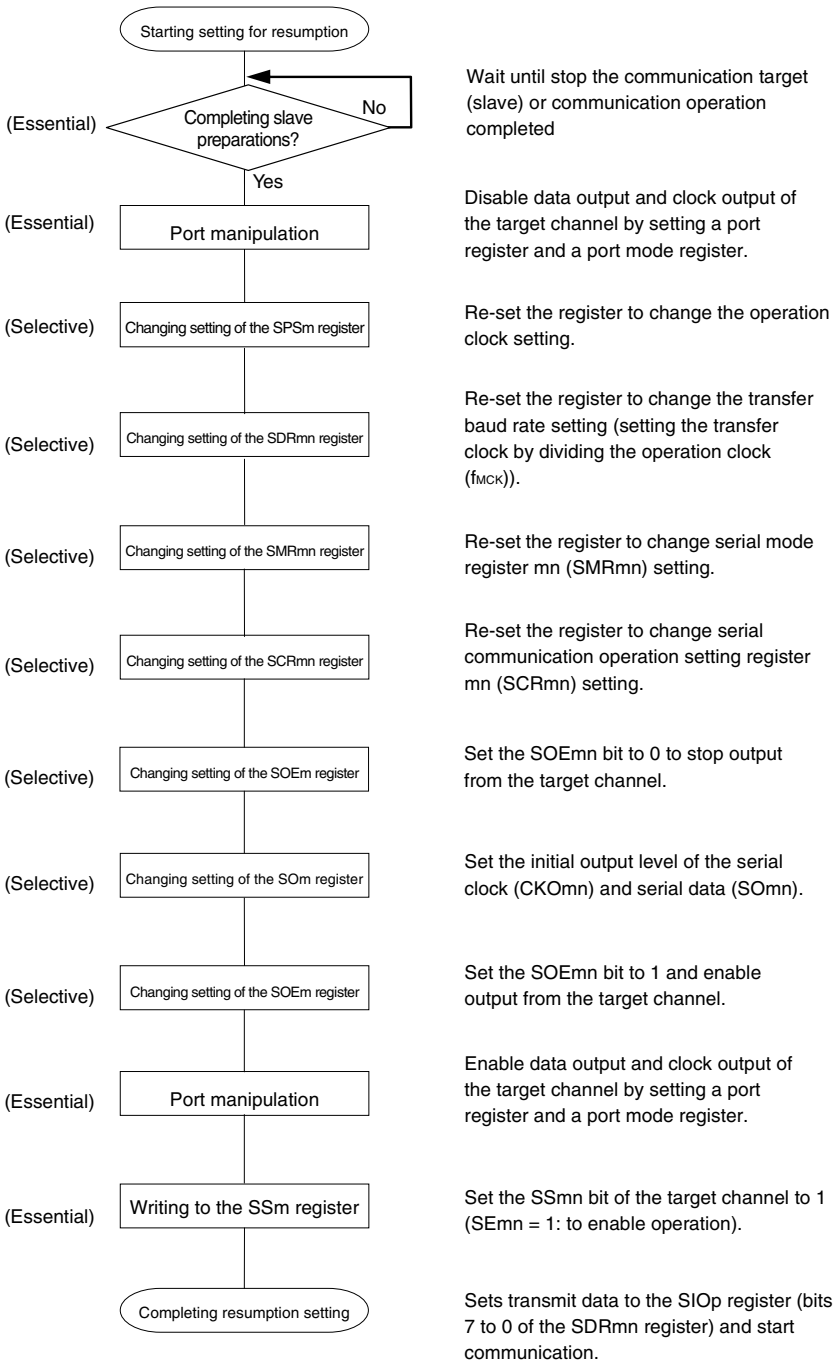
**Figure 12-42. Procedure for Stopping Master Transmission/Reception**




<R>

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

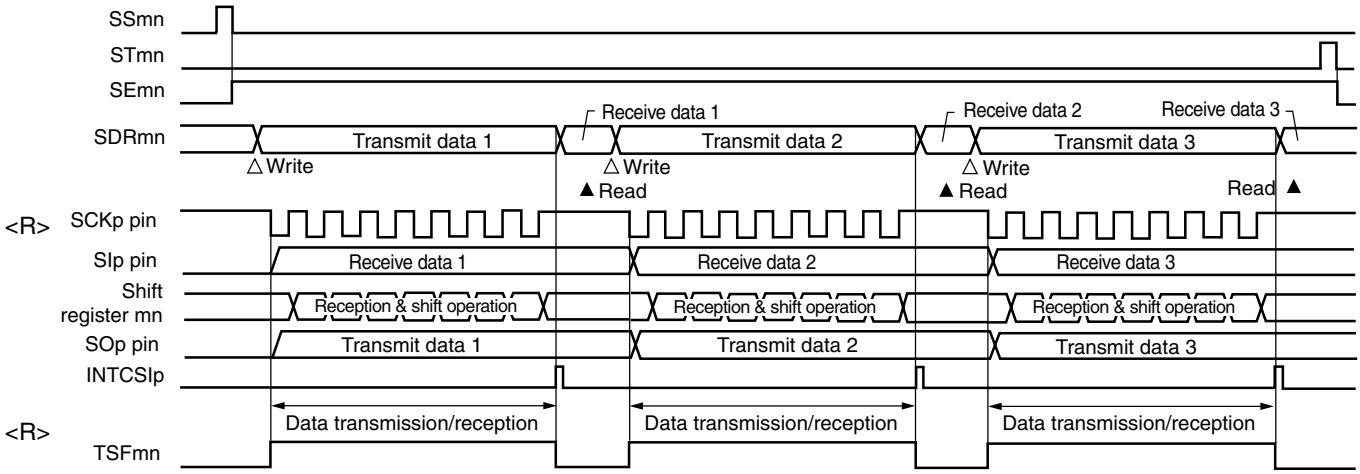
**Figure 12-43. Procedure for Resuming Master Transmission/Reception**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

### (3) Processing flow (in single-transmission/reception mode)

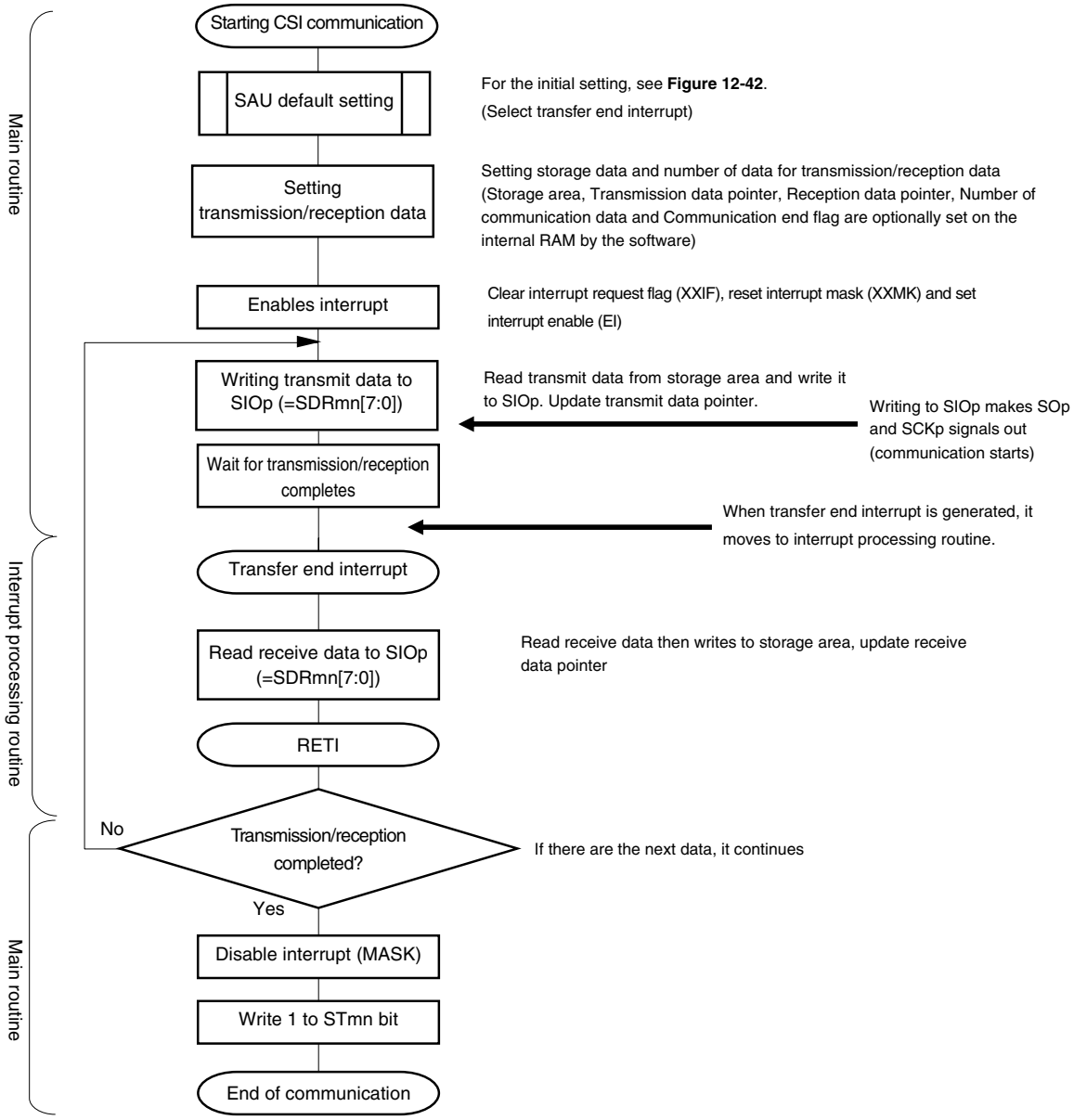
**Figure 12-44. Timing Chart of Master Transmission/Reception (in Single-Transmission/Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**Figure 12-45. Flowchart of Master Transmission/Reception (in Single-Transmission/Reception Mode)**

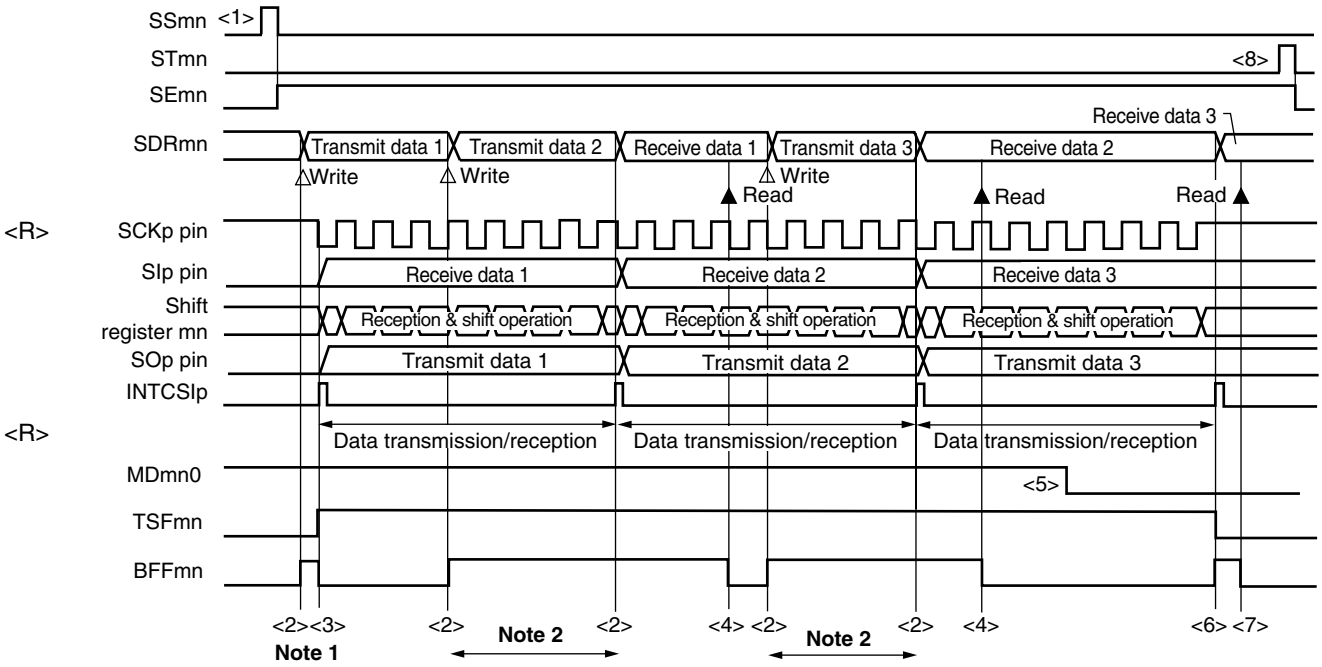
<R>




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

### (4) Processing flow (in continuous transmission/reception mode)

**Figure 12-46. Timing Chart of Master Transmission/Reception (in Continuous Transmission/Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)**




**Notes 1.** If transmit data is written to the SDRmn register while the BFFmn bit of serial status register mn (SSRmn) is 1 (valid data is stored in serial data register mn (SDRmn)), the transmit data is overwritten.

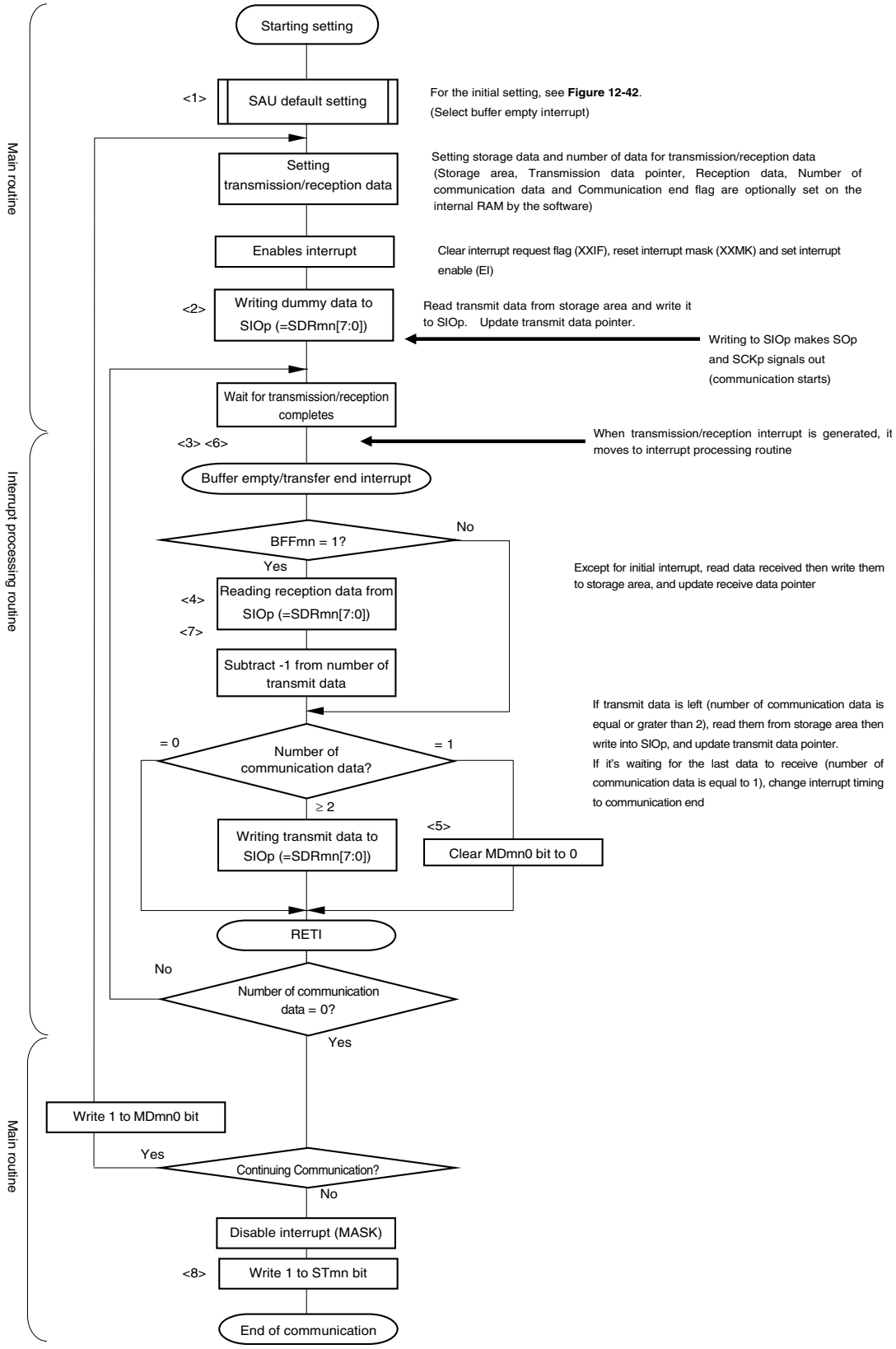
**2.** The transmit data can be read by reading the SDRmn register during this period. At this time, the transfer operation is not affected.

**Caution The MDmn0 bit of serial mode register mn (SMRmn) can be rewritten even during operation. However, rewrite it before transfer of the last bit is started, so that it has been rewritten before the transfer end interrupt of the last transmit data.**

**Remarks 1.** <1> to <8> in the figure correspond to <1> to <8> in **Figure 12-47 Flowchart of Master Transmission/Reception (in Continuous Transmission/Reception Mode)**.

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**Figure 12-47. Flowchart of Master Transmission/Reception (in Continuous Transmission/Reception Mode)**




<R>

<R>

**Remarks 1.** <1> to <8> in the figure correspond to <1> to <8> in **Figure 12-46 Timing Chart of Master Transmission/Reception (in Continuous Transmission/Reception Mode)**.

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

### 12.5.4 Slave transmission

<R> Slave transmission is that the RL78 microcontroller transmits data to another device in the state of a transfer clock being input from another device.

| 3-Wire Serial I/O | CSI00 | CSI01 | CSI10 | CSI11 | CSI20 | CSI21 |
|---|---|---|---|---|---|---|
| Target channel | Channel 0 of SAU0 | Channel 1 of SAU0 | Channel 2 of SAU0 | Channel 3 of SAU0 | Channel 0 of SAU1 | Channel 1 of SAU1 |
| Pins used | SCK00, SO00 | SCK01, SO01 | SCK10, SO10 | SCK11, SO11 | SCK20, SO20 | SCK21, SO21 |
| Interrupt | INTCSI00 | INTCSI01 | INTCSI10 | INTCSI11 | INTCSI20 | INTCSI21 |
| | Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected. | | | | | |
| Error detection flag | Overrun error detection flag (OVFmn) only | | | | | |
| Transfer data length | 7 or 8 bits | | | | | |
| Transfer rate | Max. $f_{MCK}$/6 [MHz]Notes 1, 2. | | | | | |
| Data phase | Selectable by the DAPmn bit of the SCRmn register<br>• DAPmn = 0: Data output starts from the start of the operation of the serial clock.<br>• DAPmn = 1: Data output starts half a clock before the start of the serial clock operation. | | | | | |
| Clock phase | Selectable by the CKPmn bit of the SCRmn register<br>• CKPmn = 0: Forward<br>• CKPmn = 1: Reverse | | | | | |
| Data direction | MSB or LSB first | | | | | |

**Notes 1.** Because the external serial clock input to the SCK00, SCK01, SCK10, SCK11, SCK20, and SCK21 pins is sampled internally and used, the fastest transfer rate is $f_{MCK}$/6 [MHz].

**2.** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristic in the electrical specifications (see **CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C), CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to +105°C)**).

**Remarks 1.** $f_{MCK}$: Operation clock frequency of target channel

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

### (1) Register setting

**Figure 12-48. Example of Contents of Registers for Slave Transmission of 3-Wire Serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) (1/2)**

**(a) Serial mode register mn (SMRmn)**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SMRmn | CKSmn | CCSmn | | | | | | STSmn | | SISmn0 | | | | MDmn2 | MDmn1 | MDmn0 |
| | 0/1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0/1 |

Operation clock ($f_{MCK}$) of channel n
0: Prescaler output clock CKm0 set by the SPSm register
1: Prescaler output clock CKm1 set by the SPSm register

Interrupt source of channel n
0: Transfer end interrupt
1: Buffer empty interrupt

**(b) Serial communication operation setting register mn (SCRmn)**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SCRmn | TXEmn | RXEmn | DAPmn | CKPmn | | EOCmn | PTCmn1 | PTCmn0 | DIRmn | | SLCmn1 | SLCmn0 | | | DLSmn1 | DLSmn0 |
| | 1 | 0 | 0/1 | 0/1 | 0 | 0 | 0 | 0 | 0/1 | 0 | 0 | 0 | 0 | 1 | 1 | 0/1 |

Selection of the data and clock phase (For details about the setting, see **12.3 Registers Controlling Serial Array Unit**.)

Selection of data transfer sequence
0: Inputs/outputs data with MSB first
1: Inputs/outputs data with LSB first.

Setting of data length
0: 7-bit data length
1: 8-bit data length

**(c) Serial data register mn (SDRmn) (lower 8 bits: SIOp)**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SDRmn | 0000000<br>Baud rate setting | | | | | | | 0 | Transmit data setting | | | | | | | |

SIOp

**(d) Serial output register m (SOm) … Sets only the bits of the target channel.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SOm | 0 | 0 | 0 | 0 | CKOm3 Note | CKOm2 Note | CKOm1 | CKOm0 | 0 | 0 | 0 | 0 | SOm3 Note | SOm2 Note | SOm1 | SOm0 |
| | | | | | × | × | × | × | | | | | 0/1 | 0/1 | 0/1 | 0/1 |

**Note** Unit 0 only

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**2.** ☐: Setting is fixed in the CSI slave transmission mode, ▩: Setting disabled (set to the initial value)
×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)
0/1: Set to 0 or 1 depending on the usage of the user

**Figure 12-48. Example of Contents of Registers for Slave Transmission of 3-Wire Serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) (2/2)**

**(e) Serial output enable register m (SOEm) … Sets only the bits of the target channel to 1.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SOEm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SOEm3 Note 0/1 | SOEm2 Note 0/1 | SOEm1 0/1 | SOEm0 0/1 |

**(f) Serial channel start register m (SSm) … Sets only the bits of the target channel to 1.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SSm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SSm3 Note 0/1 | SSm2 Note 0/1 | SSm1 0/1 | SSm0 0/1 |

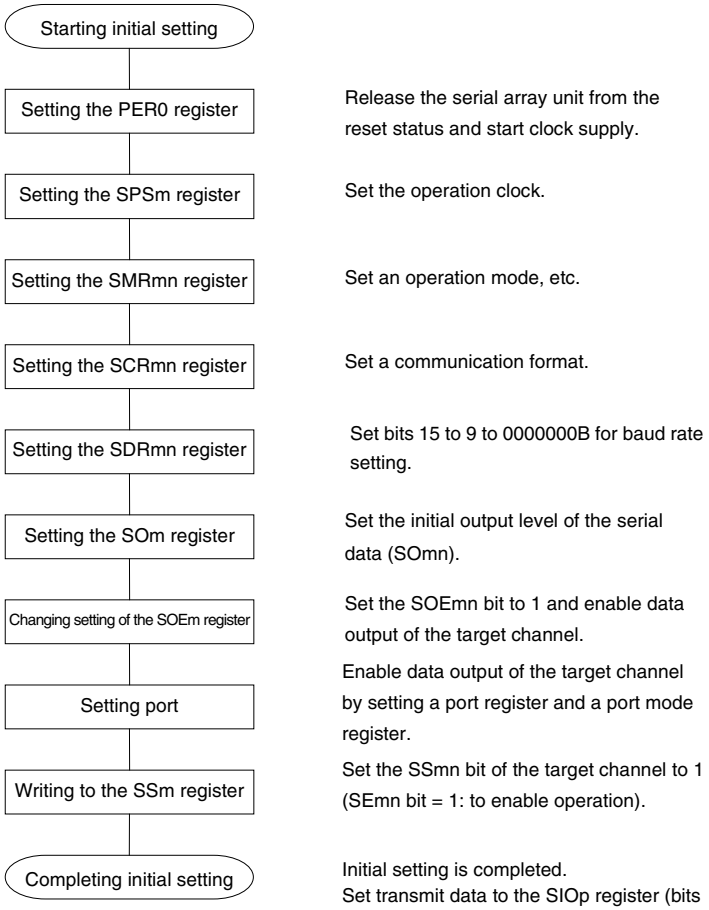
**Note** Unit 0 only

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3),
p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**2.** ☐: Setting disabled (set to the initial value)
×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)
0/1: Set to 0 or 1 depending on the usage of the user

### (2) Operation procedure

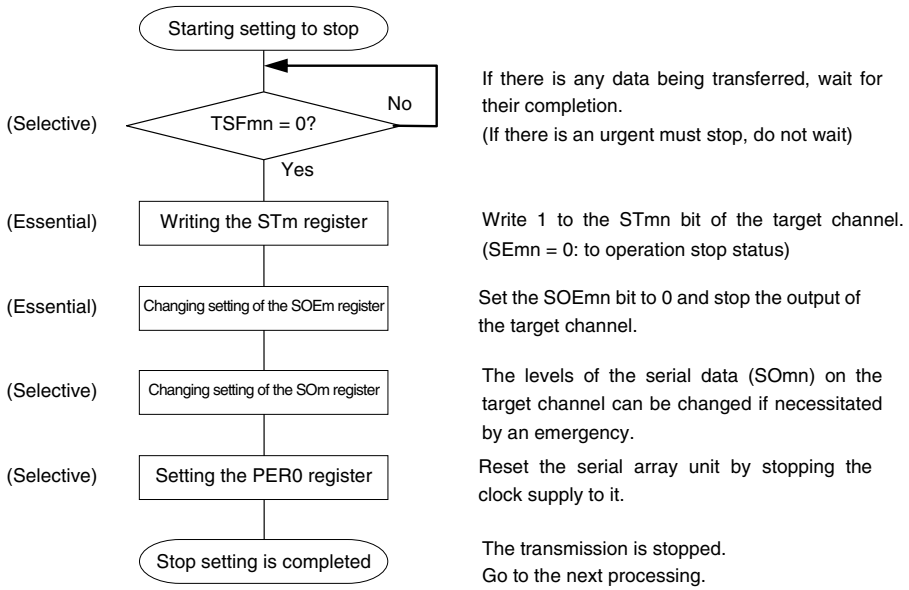
**Figure 12-49. Initial Setting Procedure for Slave Transmission**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

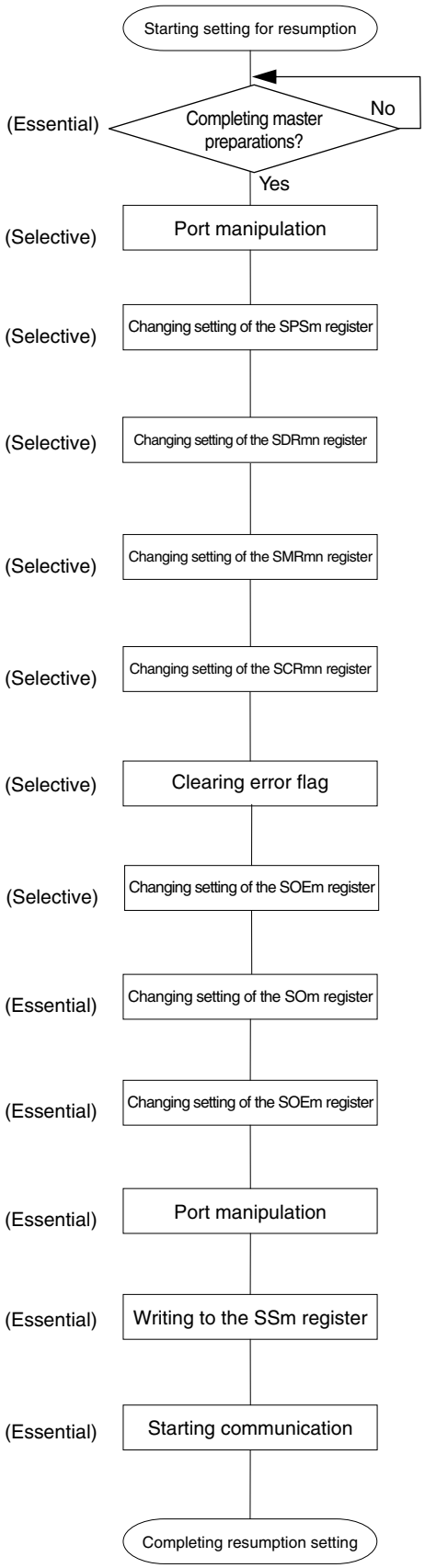
**Figure 12-50. Procedure for Stopping Slave Transmission**

<R>




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

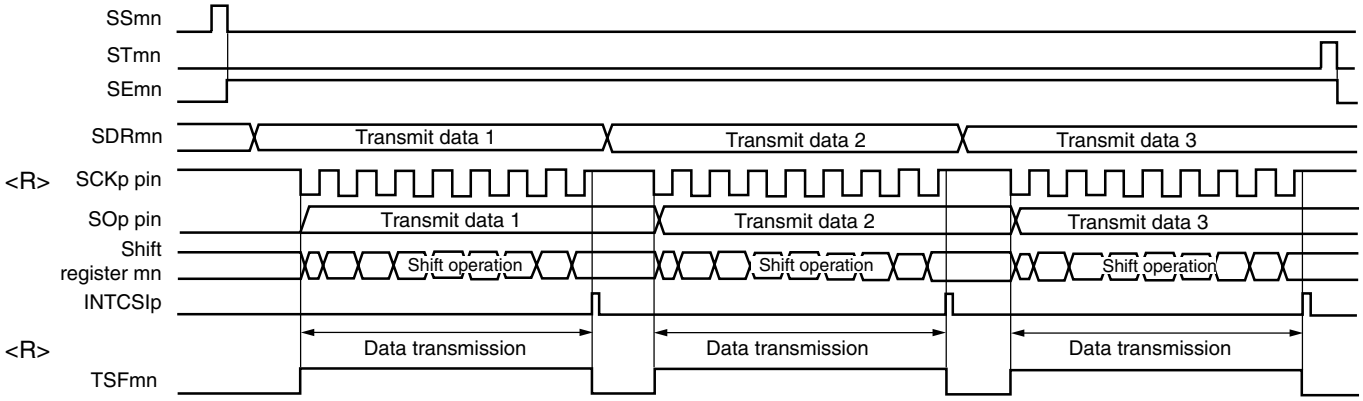
**Figure 12-51. Procedure for Resuming Slave Transmission**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

### (3) Processing flow (in single-transmission mode)

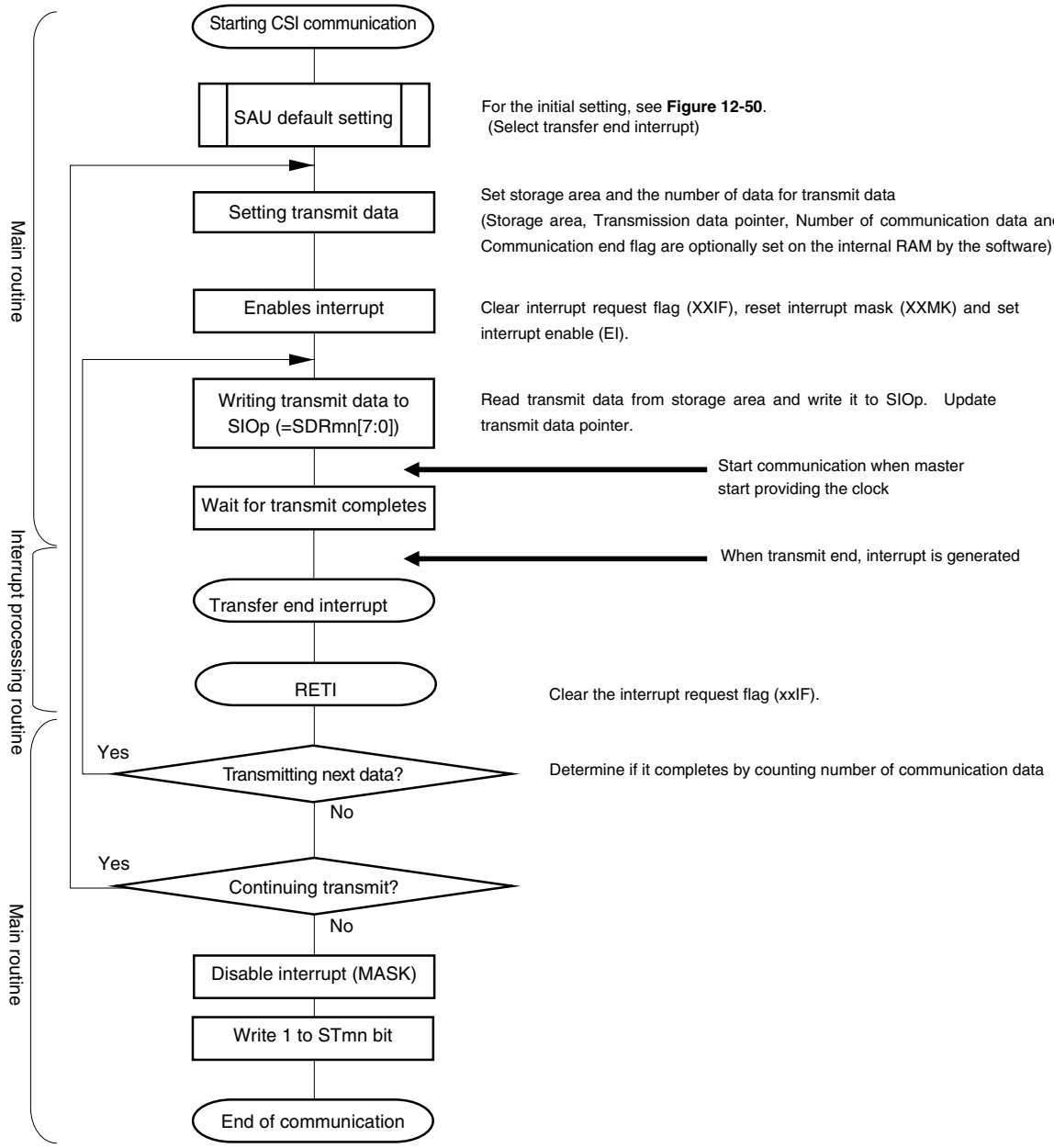
**Figure 12-52. Timing Chart of Slave Transmission (in Single-Transmission Mode)**
**(Type 1: DAPmn = 0, CKPmn = 0)**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**Figure 12-53. Flowchart of Slave Transmission (in Single-Transmission Mode)**

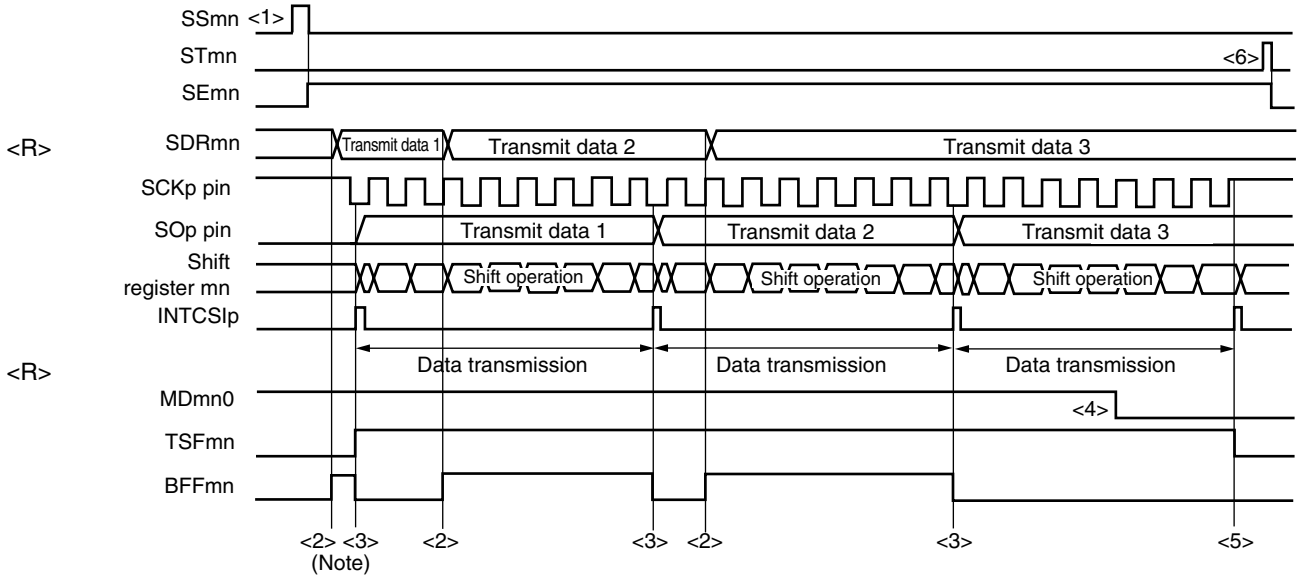
<R>




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

### (4) Processing flow (in continuous transmission mode)

**Figure 12-54. Timing Chart of Slave Transmission (in Continuous Transmission Mode)**
**(Type 1: DAPmn = 0, CKPmn = 0)**

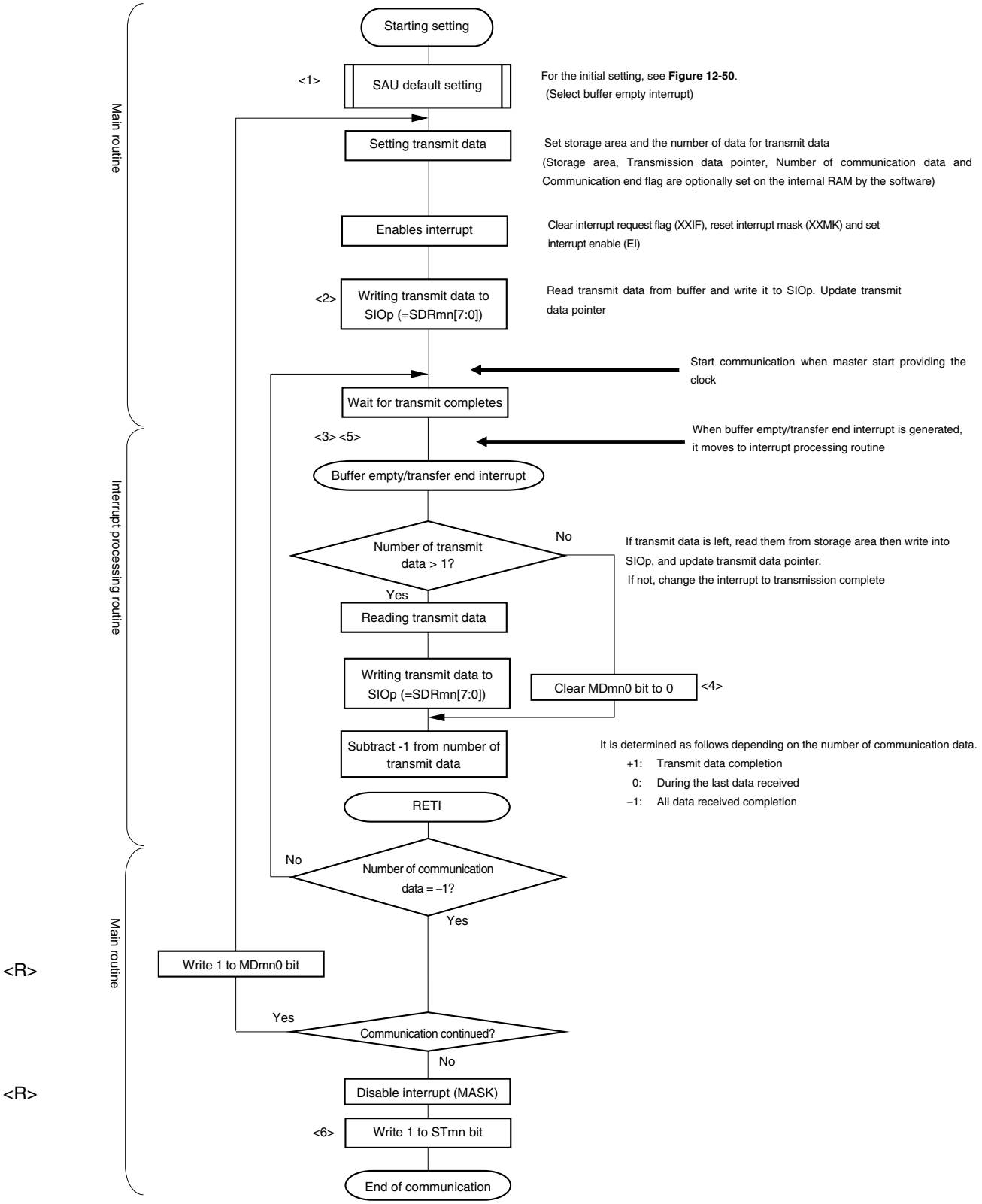



**Note** If transmit data is written to the SDRmn register while the BFFmn bit of serial status register mn (SSRmn) is 1 (valid data is stored in serial data register mn (SDRmn)), the transmit data is overwritten.

**Caution The MDmn0 bit of serial mode register mn (SMRmn) can be rewritten even during operation. However, rewrite it before transfer of the last bit is started.**

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**Figure 12-55. Flowchart of Slave Transmission (in Continuous Transmission Mode)**




<R>

<R>

**Remarks 1.** <1> to <6> in the figure correspond to <1> to <6> in **Figure 12-54 Timing Chart of Slave Transmission (in Continuous Transmission Mode)**.

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

### 12.5.5 Slave reception

<R> Slave reception is that the RL78 microcontroller receives data from another device in the state of a transfer clock being input from another device.

| 3-Wire Serial I/O | CSI00 | CSI01 | CSI10 | CSI11 | CSI20 | CSI21 |
|---|---|---|---|---|---|---|
| Target channel | Channel 0 of SAU0 | Channel 1 of SAU0 | Channel 2 of SAU0 | Channel 3 of SAU0 | Channel 0 of SAU1 | Channel 1 of SAU1 |
| Pins used | SCK00, SI00 | SCK01, SI01 | SCK10, SI10 | SCK11, SI11 | SCK20, SI20 | SCK21, SI21 |
| Interrupt | INTCSI00 | INTCSI01 | INTCSI10 | INTCSI11 | INTCSI20 | INTCSI21 |
| | Transfer end interrupt only (Setting the buffer empty interrupt is prohibited.) | | | | | |
| Error detection flag | Overrun error detection flag (OVFmn) only | | | | | |
| Transfer data length | 7 or 8 bits | | | | | |
| Transfer rate | Max. $f_{MCK}$/6 [MHz]Notes 1, 2 | | | | | |
| Data phase | Selectable by the DAPmn bit of the SCRmn register<br>• DAPmn = 0: Data input starts from the start of the operation of the serial clock.<br>• DAPmn = 1: Data input starts half a clock before the start of the serial clock operation. | | | | | |
| Clock phase | Selectable by the CKPmn bit of the SCRmn register<br>• CKPmn = 0: Non-reverse<br>• CKPmn = 1: Reverse | | | | | |
| Data direction | MSB or LSB first | | | | | |

**Notes 1.** Because the external serial clock input to the SCK00, SCK01, SCK10, SCK11, SCK20, and SCK21 pins is sampled internally and used, the fastest transfer rate is $f_{MCK}$/6 [MHz].

**2.** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristic in the electrical specifications (see **CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C), CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to +105°C)**).

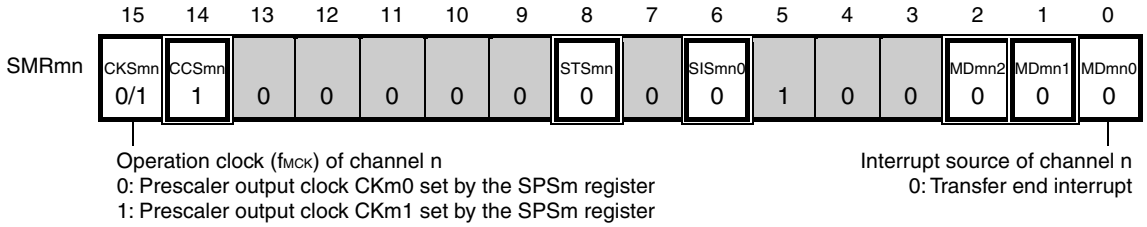
**Remarks 1.** $f_{MCK}$: Operation clock frequency of target channel

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

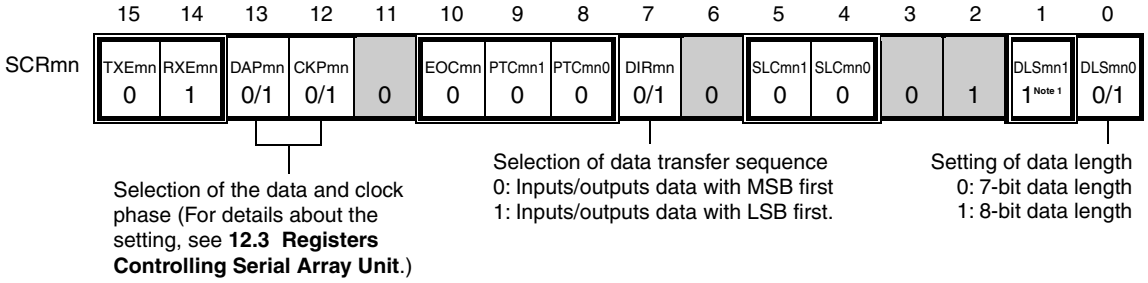
### (1) Register setting

**Figure 12-56. Example of Contents of Registers for Slave Reception of 3-Wire Serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21)**

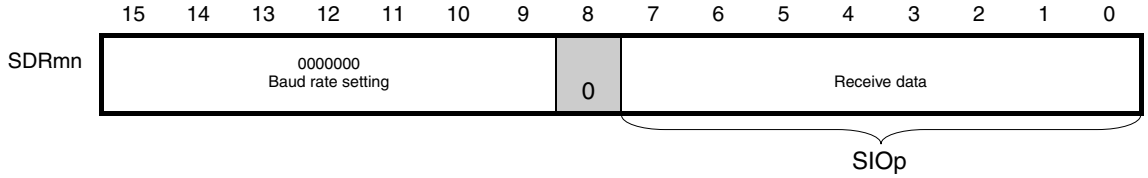
**(a) Serial mode register mn (SMRmn)**



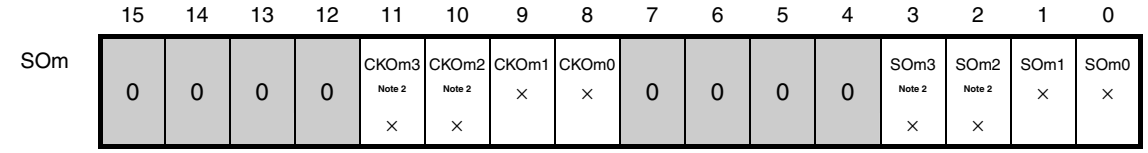

**(b) Serial communication operation setting register mn (SCRmn)**




**(c) Serial data register mn (SDRmn) (lower 8 bits: SIOp)**




**(d) Serial output register m (SOm) …The Register that not used in this mode.**




**Notes 1.** Only provided for the SCR00 and SCR01 registers. This bit is fixed to 1 for the other registers.

**2.** Unit 0 only

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3),
p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**2.** ☐: Setting is fixed in the CSI slave transmission mode, ▒: Setting disabled (set to the initial value)
×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)
0/1: Set to 0 or 1 depending on the usage of the user

**Figure 12-56. Example of Contents of Registers for Slave Reception of 3-Wire Serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) (2/2)**

**(e) Serial output enable register m (SOEm) …The Register that not used in this mode.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SOEm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SOEm3 Note × | SOEm2 Note × | SOEm1 × | SOEm0 × |

**(f) Serial channel start register m (SSm) … Sets only the bits of the target channel to 1.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SSm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SSm3 Note 0/1 | SSm2 Note 0/1 | SSm1 0/1 | SSm0 0/1 |

**Note** Unit 0 only

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

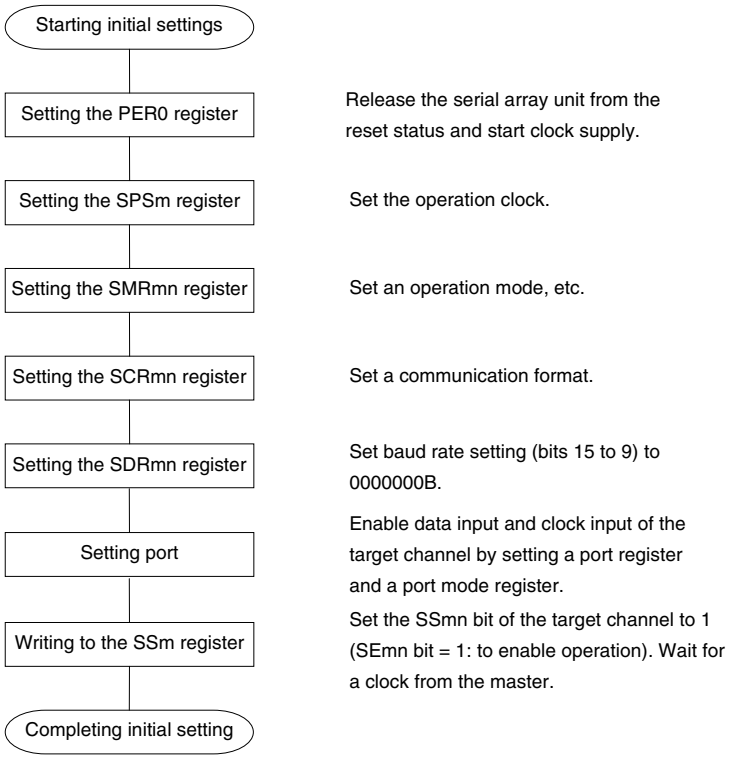
**2.** ☐ : Setting disabled (set to the initial value)

×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)

0/1: Set to 0 or 1 depending on the usage of the user

### (2) Operation procedure

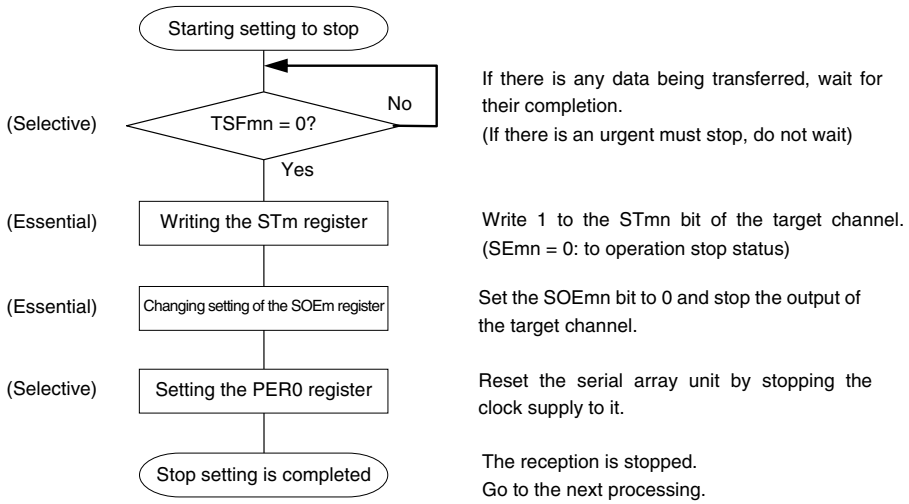
**Figure 12-57. Initial Setting Procedure for Slave Reception**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

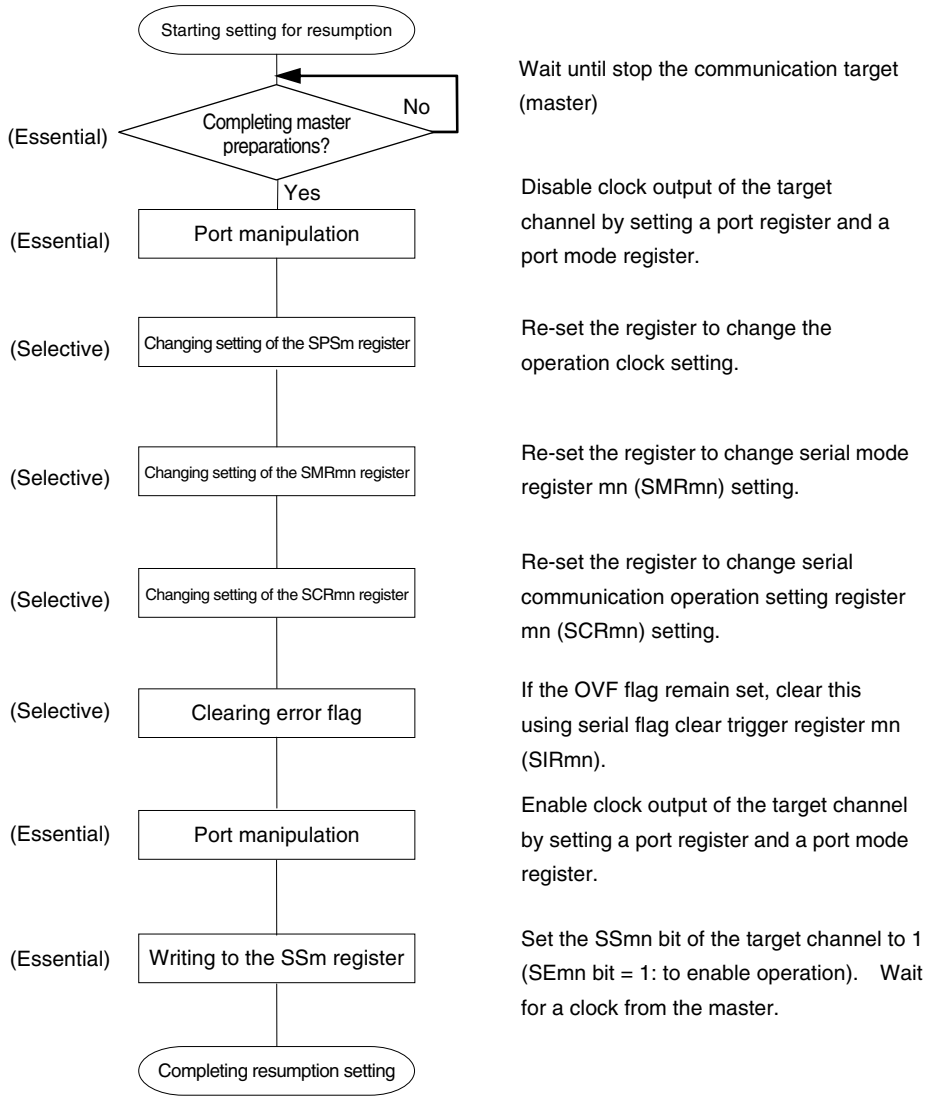
**Figure 12-58. Procedure for Stopping Slave Reception**

<R>




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

**Figure 12-59. Procedure for Resuming Slave Reception**

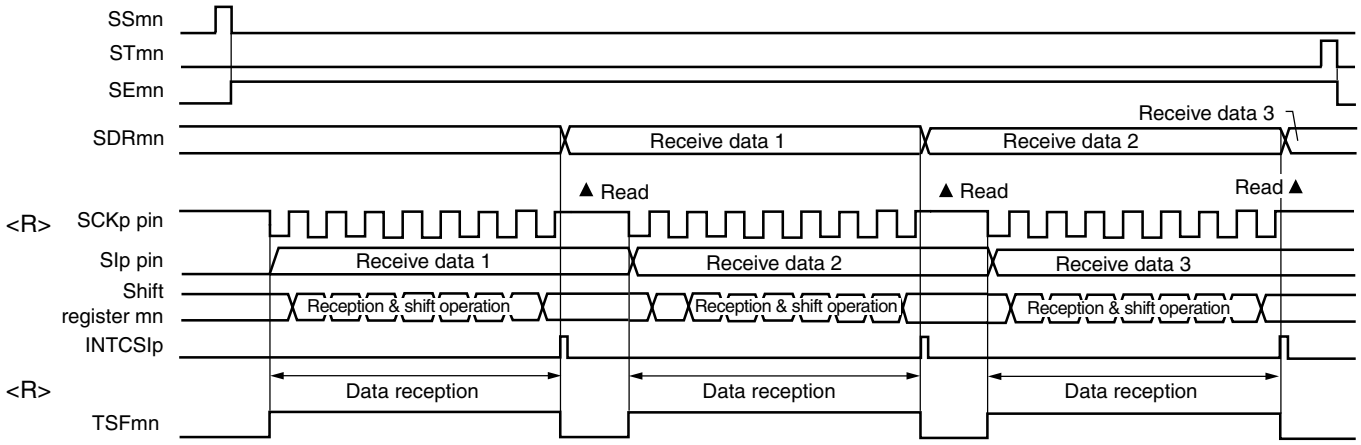



**Remarks 1.** If PER0 is rewritten while stopping the master transmission and the clock supply is stopped, wait until the transmission target (master) stops or transmission finishes, and then perform initialization instead of restarting the transmission.

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

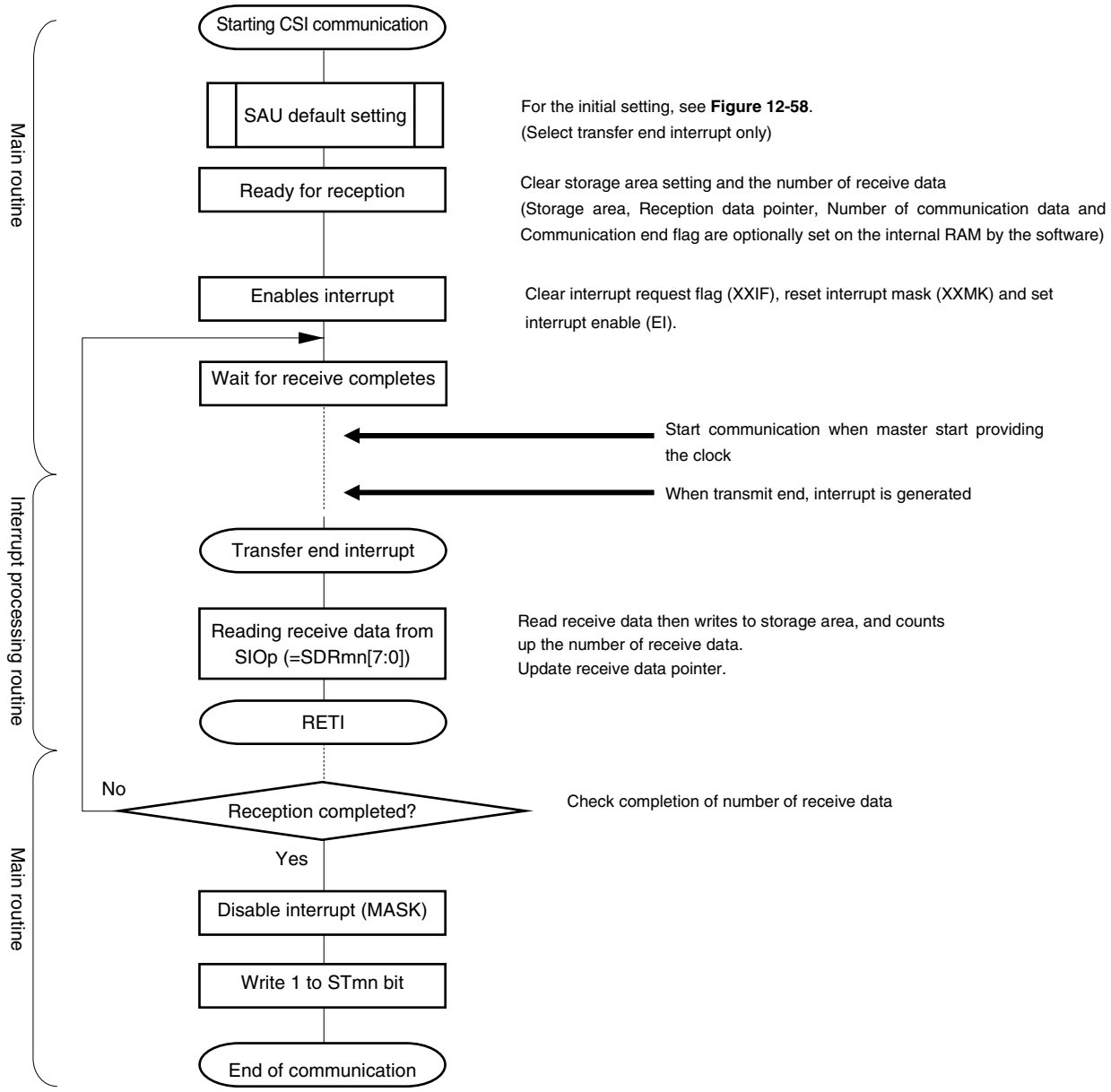
### (3) Processing flow (in single-reception mode)

**Figure 12-60. Timing Chart of Slave Reception (in Single-Reception Mode)**
**(Type 1: DAPmn = 0, CKPmn = 0)**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**Figure 12-61. Flowchart of Slave Reception (in Single-Reception Mode)**




<R>

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

### 12.5.6 Slave transmission/reception

<R> Slave transmission/reception is that the RL78 microcontroller transmits/receives data to/from another device in the state of a transfer clock being input from another device.

| 3-Wire Serial I/O | CSI00 | CSI01 | CSI10 | CSI11 | CSI20 | CSI21 |
|---|---|---|---|---|---|---|
| Target channel | Channel 0 of SAU0 | Channel 1 of SAU0 | Channel 2 of SAU0 | Channel 3 of SAU0 | Channel 0 of SAU1 | Channel 1 of SAU1 |
| Pins used | SCK00, SI00, SO00 | SCK01, SI01, SO01 | SCK10, SI10, SO10 | SCK11, SI11, SO11 | SCK20, SI20, SO20 | SCK21, SI21, SO21 |
| Interrupt | INTCSI00 | INTCSI01 | INTCSI10 | INTCSI11 | INTCSI20 | INTCSI21 |
| | Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected. | | | | | |
| Error detection flag | Overrun error detection flag (OVFmn) only | | | | | |
| Transfer data length | 7 or 8 bits | | | | | |
| Transfer rate | Max. $f_{MCK}$/6 [MHz]Notes 1, 2. | | | | | |
| Data phase | Selectable by the DAPmn bit of the SCRmn register<br>• DAPmn = 0: Data I/O starts from the start of the operation of the serial clock.<br>• DAPmn = 1: Data I/O starts half a clock before the start of the serial clock operation. | | | | | |
| Clock phase | Selectable by the CKPmn bit of the SCRmn register<br>• CKPmn = 0: Non-reverse<br>• CKPmn = 1: Reverse | | | | | |
| Data direction | MSB or LSB first | | | | | |

**Notes 1.** Because the external serial clock input to the SCK00, SCK01, SCK10, SCK11, SCK20, and SCK21 pins is sampled internally and used, the fastest transfer rate is $f_{MCK}$/6 [MHz].

**2.** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristic in the electrical specifications (see **CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C), CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to +105°C)**).

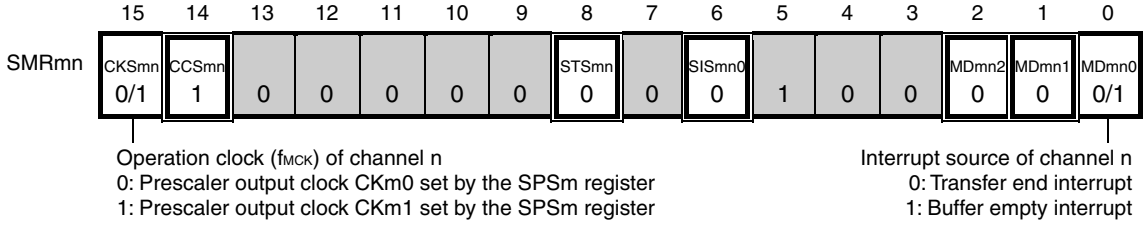
**Remarks 1.** $f_{MCK}$: Operation clock frequency of target channel

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

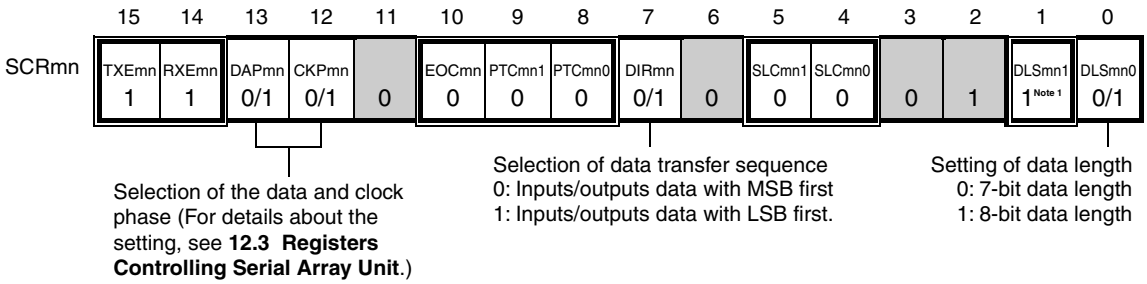
### (1) Register setting

**Figure 12-62. Example of Contents of Registers for Slave Transmission/Reception of 3-Wire Serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) (1/2)**

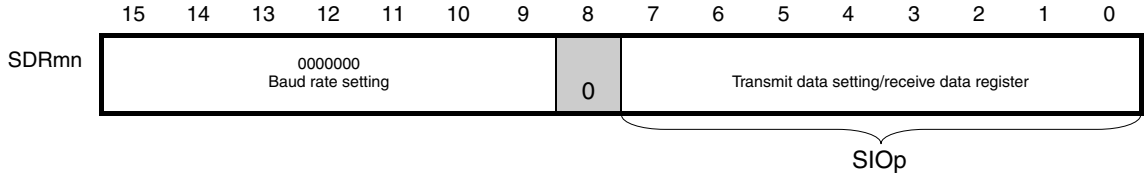
**(a) Serial mode register mn (SMRmn)**



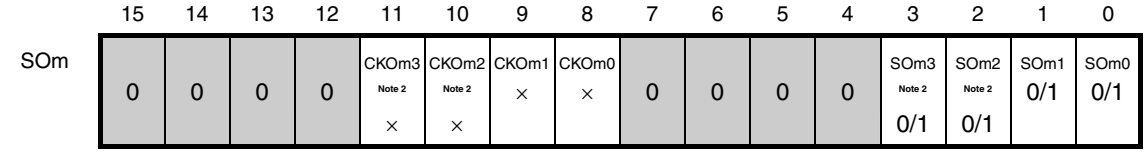

**(b) Serial communication operation setting register mn (SCRmn)**




**(c) Serial data register mn (SDRmn) (lower 8 bits: SIOp)**




**(d) Serial output register m (SOm) … Sets only the bits of the target channel.**




**Notes 1.** Only provided for the SCR00 and SCR01 registers. This bit is fixed to 1 for the other registers.

**2.** Unit 0 only

**Caution Be sure to set transmit data to the SIOp register before the clock from the master is started.**

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**2.** □: Setting is fixed in the CSI slave transmission/reception mode,

■: Setting disabled (set to the initial value)

×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)

0/1: Set to 0 or 1 depending on the usage of the user

**Figure 12-62. Example of Contents of Registers for Slave Transmission/Reception of 3-Wire Serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21, CSI30, CSI31) (2/2)**

**(e) Serial output enable register m (SOEm) … Sets only the bits of the target channel to 1.**

SOEm

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SOEm3 Note 0/1 | SOEm2 Note 0/1 | SOEm1 0/1 | SOEm0 0/1 |

**(f) Serial channel start register m (SSm) … Sets only the bits of the target channel to 1.**

SSm

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SSm3 Note 0/1 | SSm2 Note 0/1 | SSm1 0/1 | SSm0 0/1 |

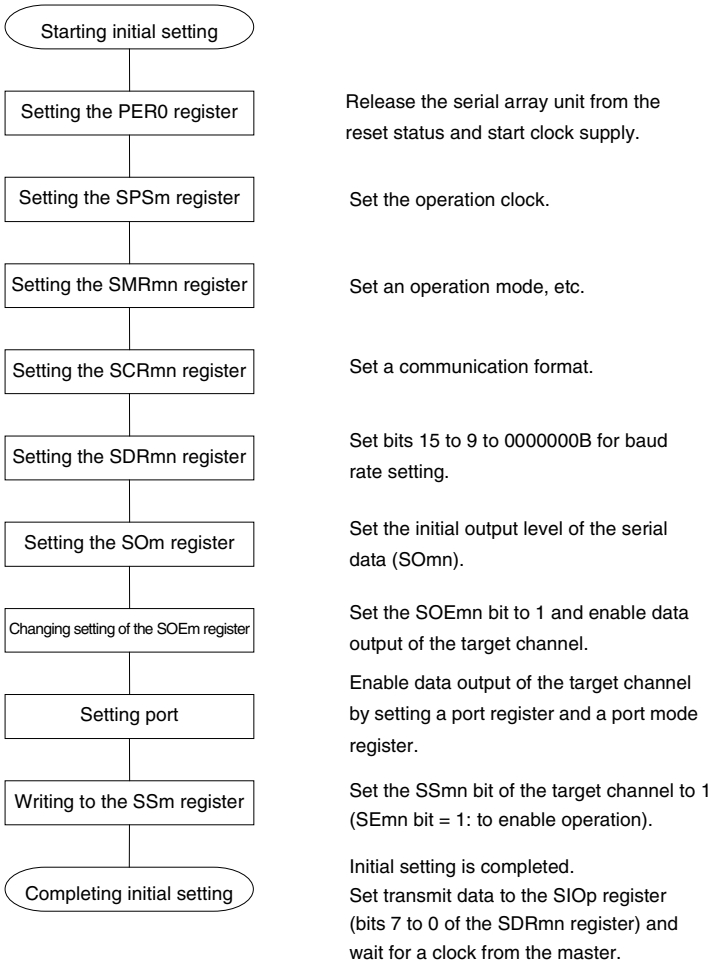
**Note** Unit 0 only

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3),
p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**2.** ☐ : Setting disabled (set to the initial value)
×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)
0/1: Set to 0 or 1 depending on the usage of the user

### (2) Operation procedure

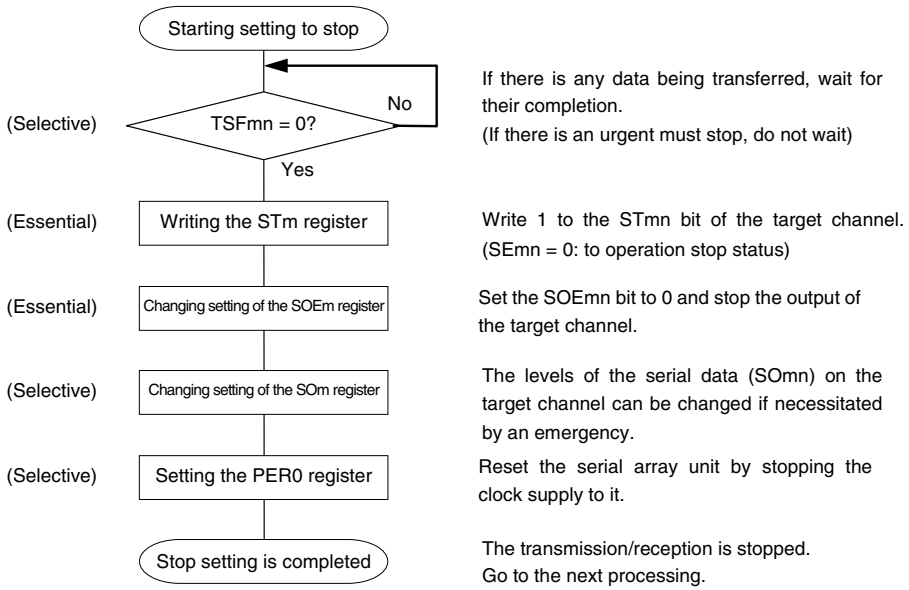
**Figure 12-63. Initial Setting Procedure for Slave Transmission/Reception**




**Caution Be sure to set transmit data to the SIOp register before the clock from the master is started.**

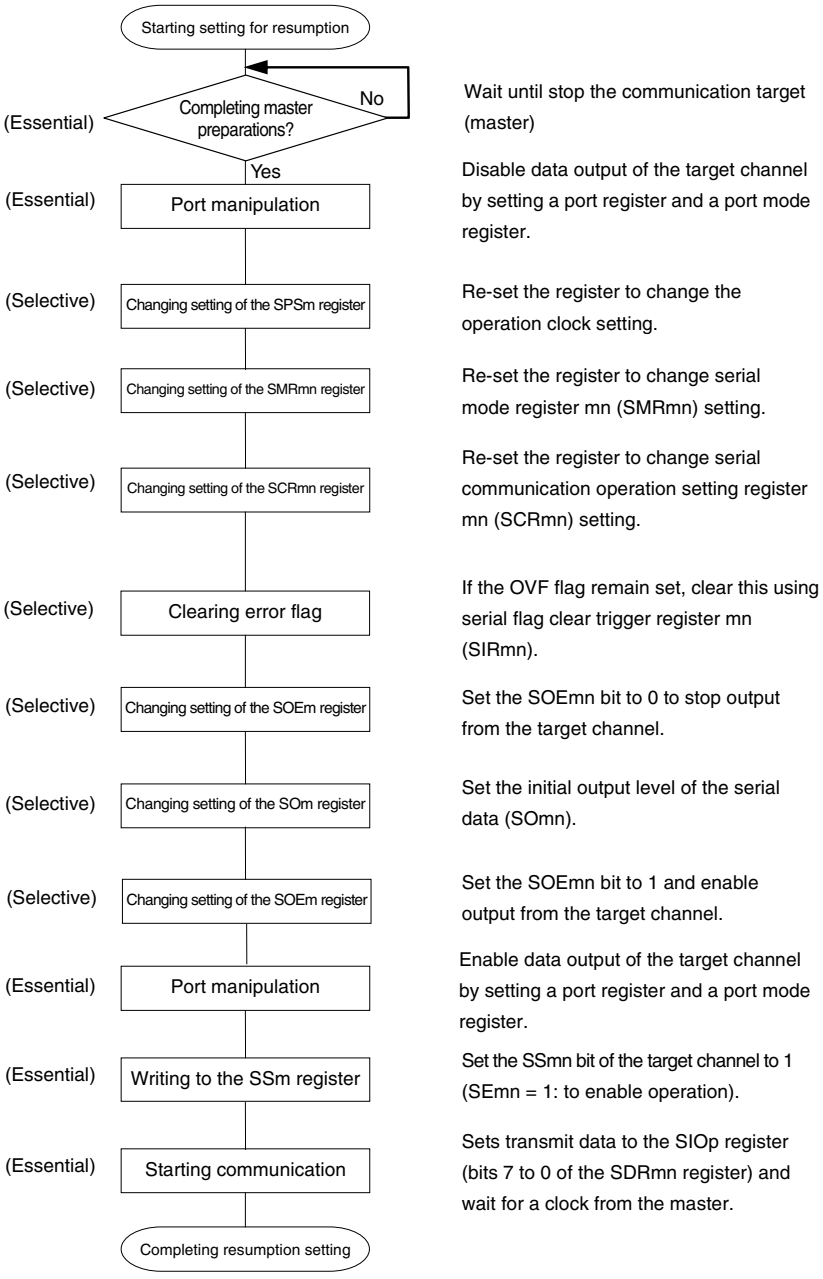
**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**Figure 12-64. Procedure for Stopping Slave Transmission/Reception**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

**Figure 12-65. Procedure for Resuming Slave Transmission/Reception**



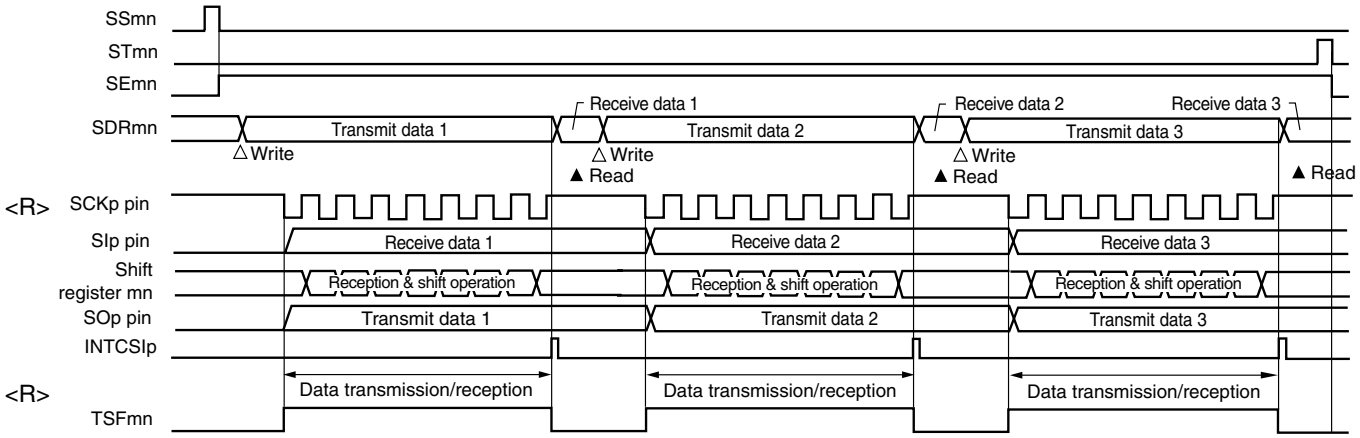

**Cautions 1. Be sure to set transmit data to the SIOp register before the clock from the master is started.**

**2. If PER0 is rewritten while stopping the master transmission and the clock supply is stopped, wait until the transmission target (master) stops or transmission finishes, and then perform initialization instead of restarting the transmission.**

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

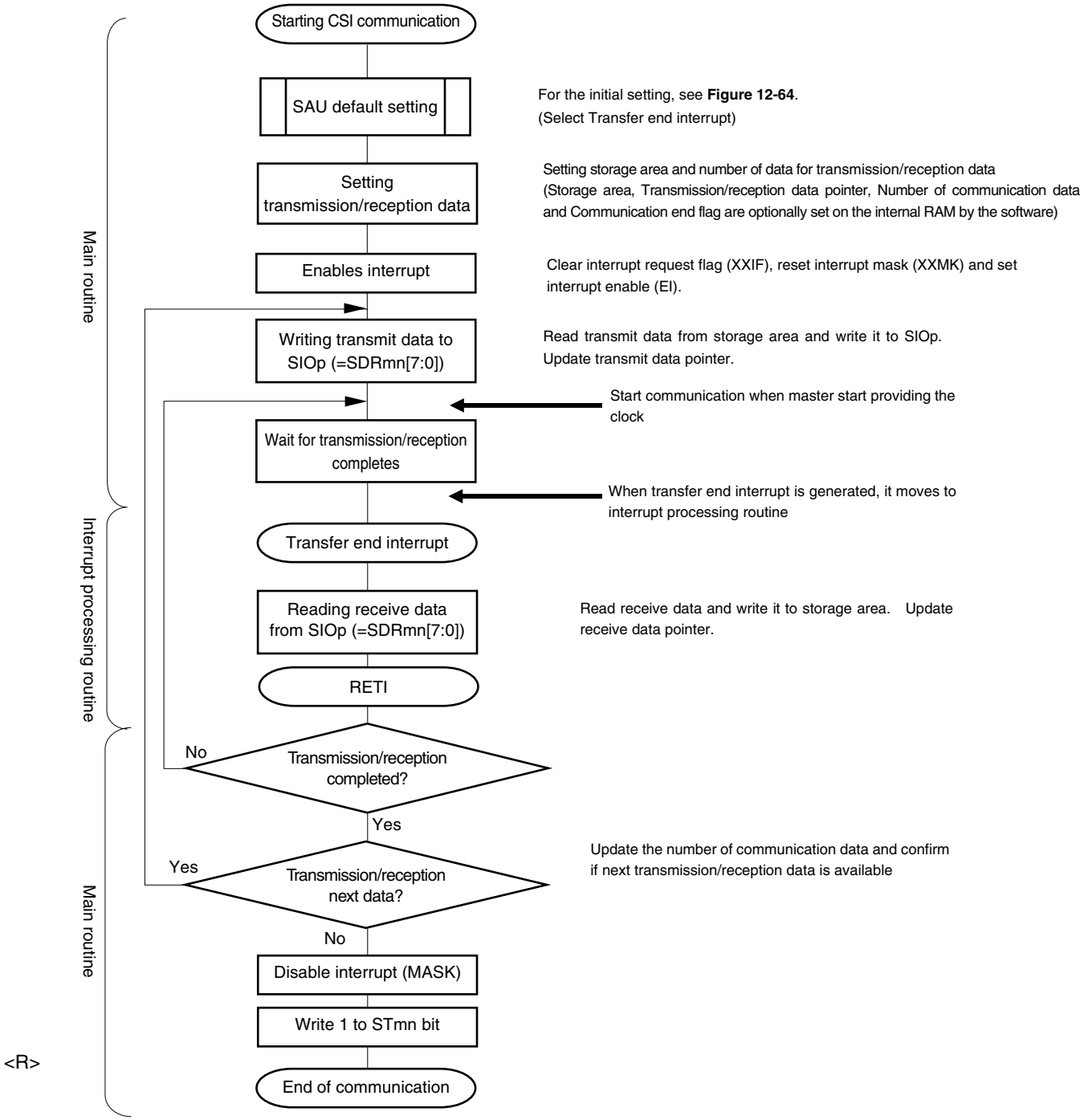
### (3) Processing flow (in single-transmission/reception mode)

**Figure 12-66. Timing Chart of Slave Transmission/Reception (in Single-Transmission/Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**Figure 12-67. Flowchart of Slave Transmission/Reception (in Single-Transmission/Reception Mode)**



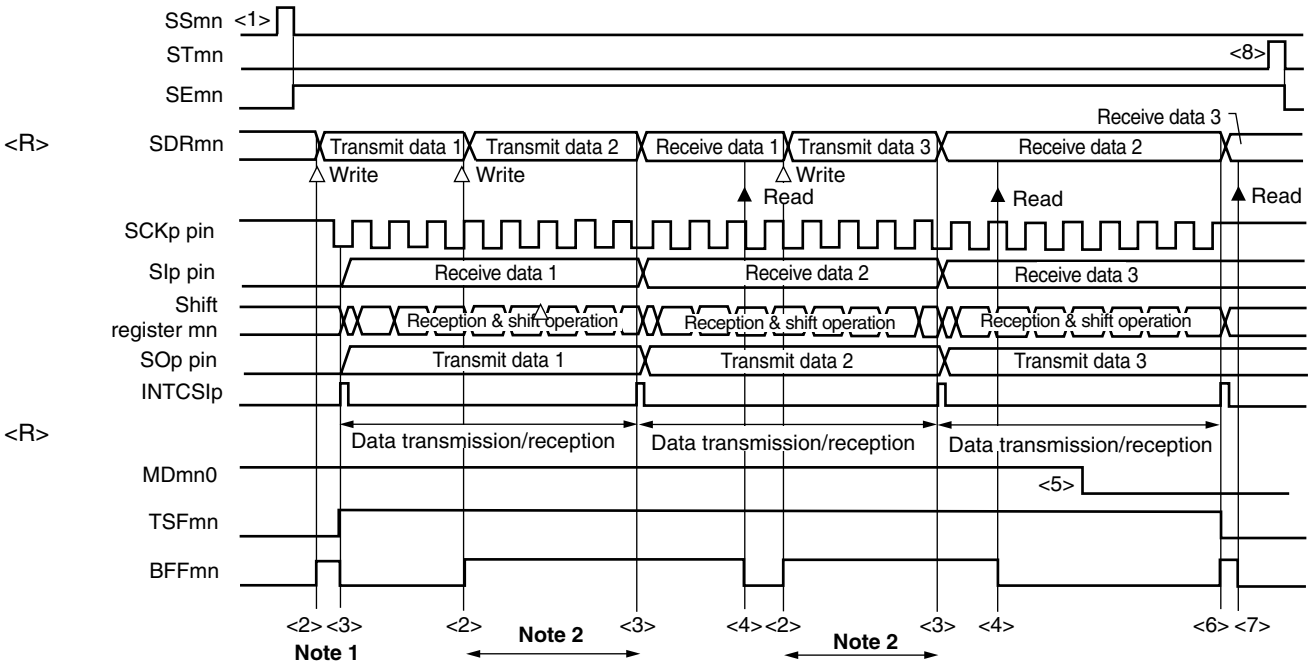

<R>

**Caution Be sure to set transmit data to the SIOp register before the clock from the master is started.**

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

### (4) Processing flow (in continuous transmission/reception mode)

**Figure 12-68. Timing Chart of Slave Transmission/Reception (in Continuous Transmission/Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)**




**Notes 1.** If transmit data is written to the SDRmn register while the BFFmn bit of serial status register mn (SSRmn) is 1 (valid data is stored in serial data register mn (SDRmn)), the transmit data is overwritten.

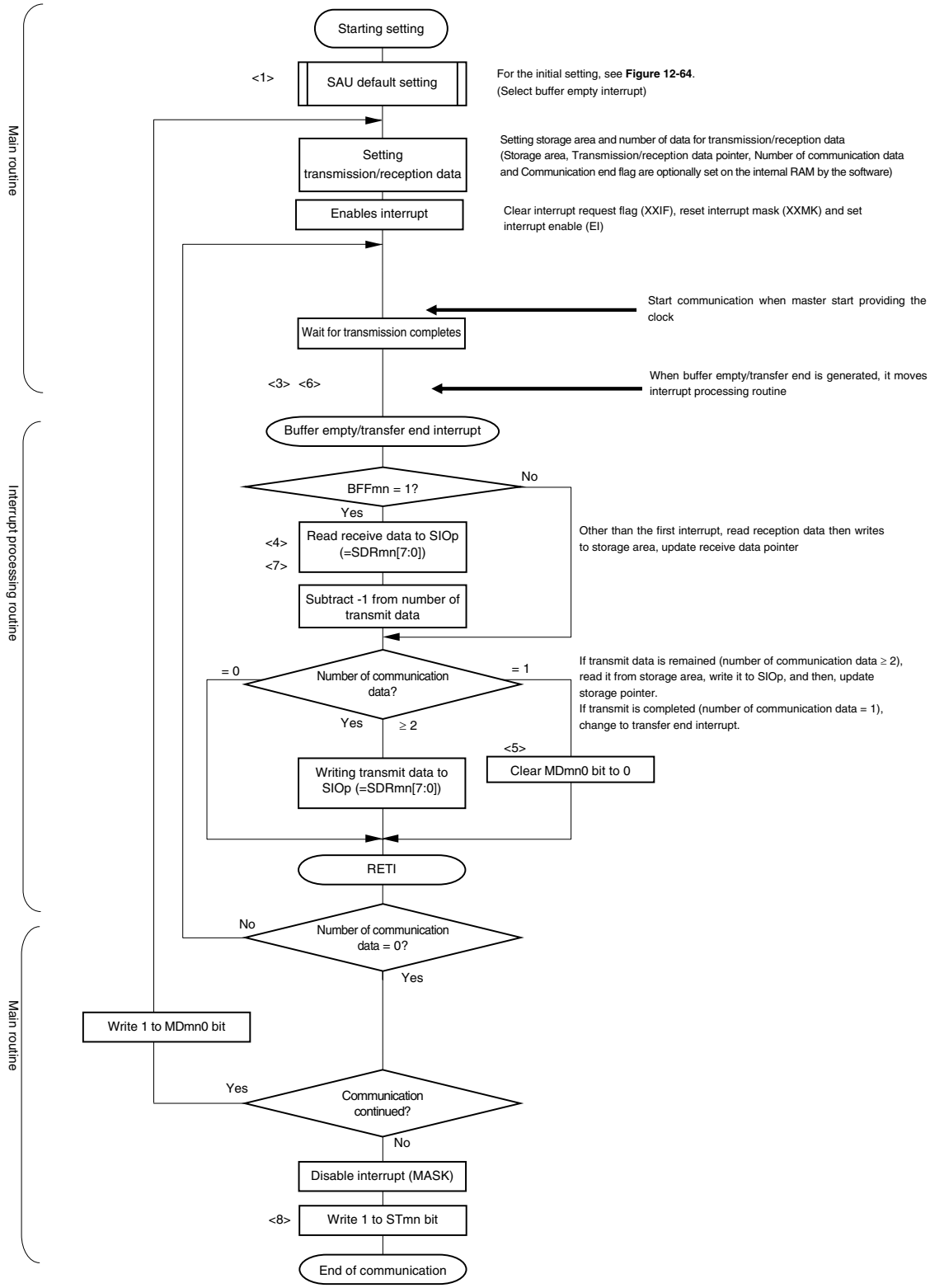
**2.** The transmit data can be read by reading the SDRmn register during this period. At this time, the transfer operation is not affected.

**Caution The MDmn0 bit of serial mode register mn (SMRmn) can be rewritten even during operation. However, rewrite it before transfer of the last bit is started, so that it has been rewritten before the transfer end interrupt of the last transmit data.**

**Remarks 1.** <1> to <8> in the figure correspond to <1> to <8> in **Figure 12-69 Flowchart of Slave Transmission/Reception (in Continuous Transmission/Reception Mode)**.

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**Figure 12-69. Flowchart of Slave Transmission/Reception (in Continuous Transmission/Reception Mode)**




<R>

**Caution Be sure to set transmit data to the SIOp register before the clock from the master is started.**

**Remarks 1.** <1> to <8> in the figure correspond to <1> to <8> in **Figure 12-68 Timing Chart of Slave Transmission/Reception (in Continuous Transmission/Reception Mode)**.

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), p: CSI number (p = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

### 12.5.7 SNOOZE mode function

SNOOZE mode makes CSI operate reception by SCKp pin input detection while the STOP mode. Normally CSI stops communication in the STOP mode. But, using the SNOOZE mode makes reception CSI operate unless the CPU operation by detecting SCKp pin input. Only CSIs00 can be specified for asynchronous reception.

When using the CSI in SNOOZE mode, make the following setting before switching to the STOP mode (see **Figure 12-72 Flowchart of SNOOZE Mode Operation (Once Startup)** and **Figure 12-74 Flowchart of SNOOZE Mode Operation (Continuous Startup)**).

- When using the SNOOZE mode function, set the SWCm bit of serial standby control register m (SSCm) to 1 just before switching to the STOP mode. After the initial setting has been completed, set the SSm1 bit of serial channel start register m (SSm) to 1.

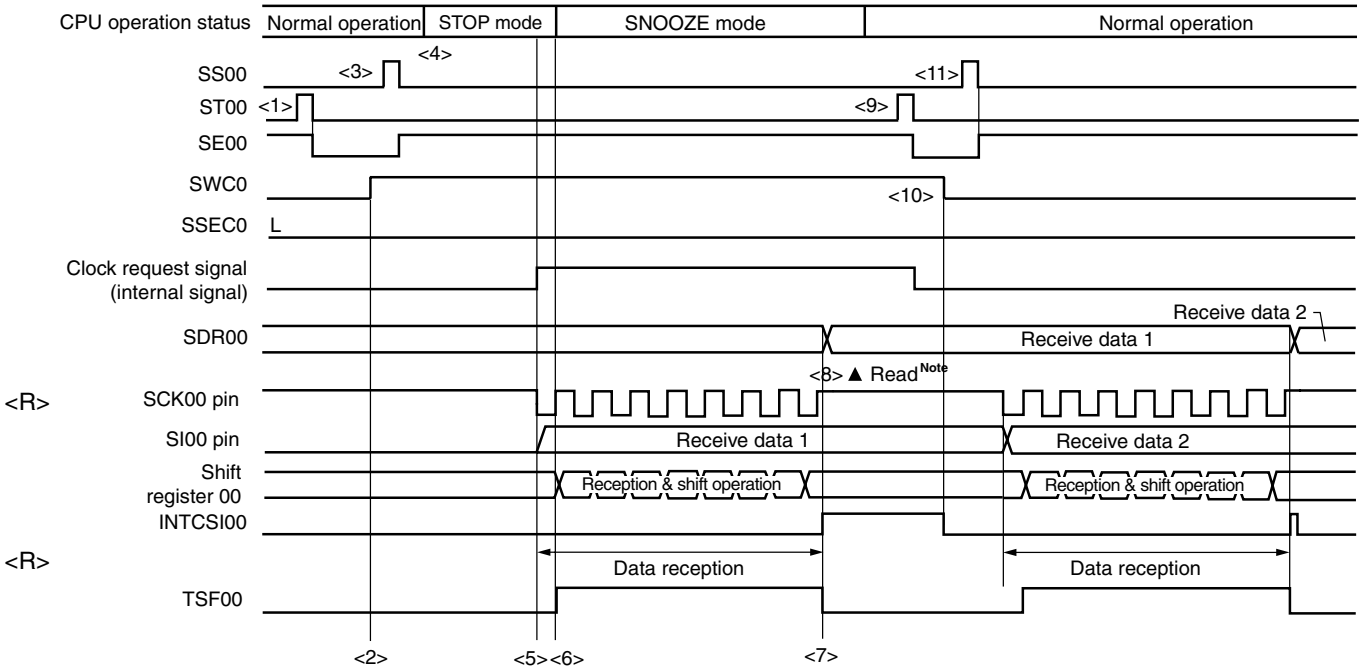
After a transition to the STOP mode, the CSI starts reception operations upon detection of an edge of the SCKp pin.

**Cautions 1. The SNOOZE mode can only be specified when the high-speed on-chip oscillator clock is selected for $f_{CLK}$.**

**2. The maximum transfer rate when using CSIp in the SNOOZE mode is 1 Mbps.**

**(1) SNOOZE mode operation (once startup)**

**Figure 12-70. Timing Chart of SNOOZE Mode Operation (Once Startup) (Type 1: DAPmn = 0, CKPmn = 0)**




**Note** Only read received data while SWCm = 1 and before the next edge of the SCKp pin input is detected.

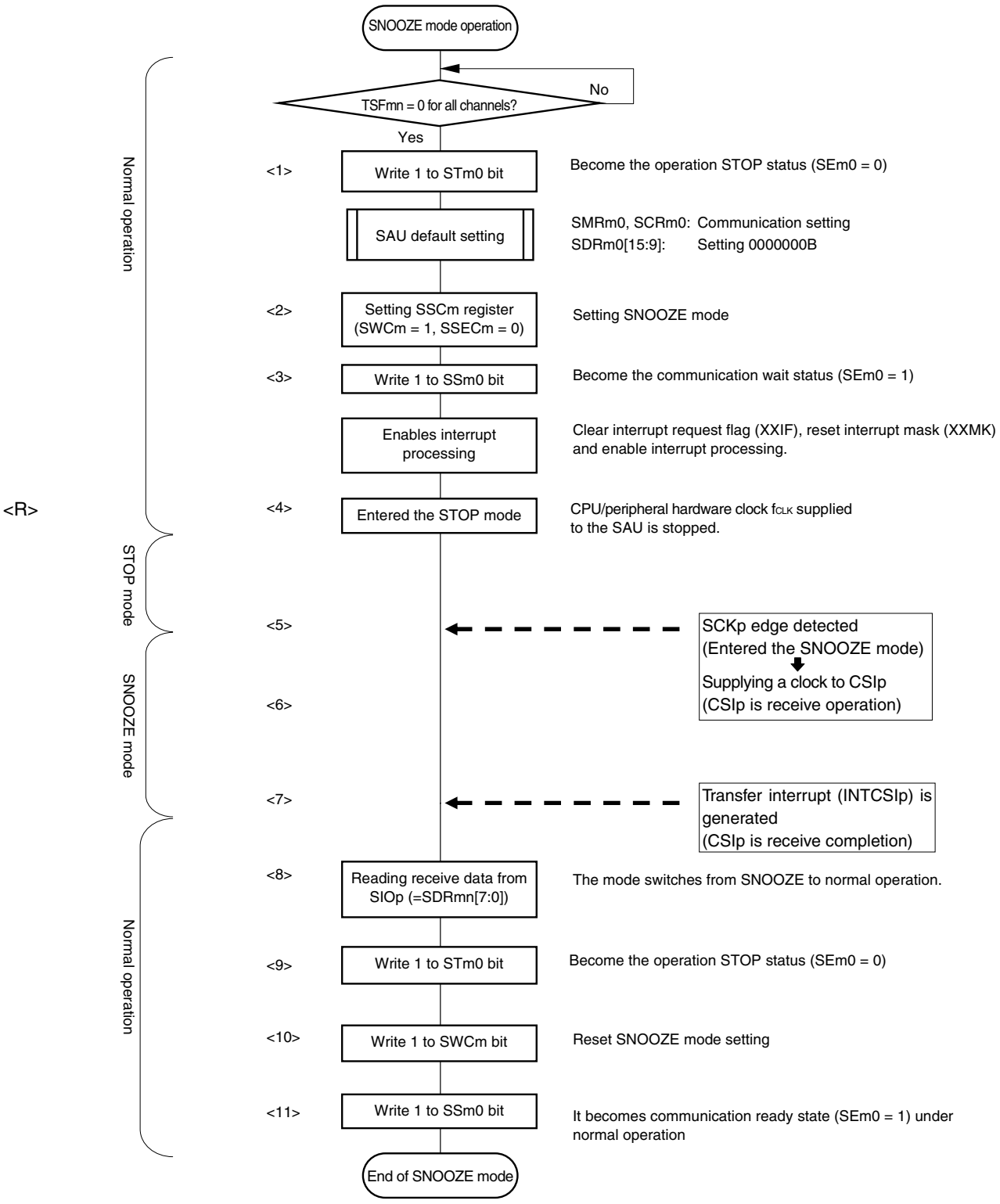
**Cautions 1. Before switching to the SNOOZE mode or after reception operation in the SNOOZE mode finishes, set the STm0 bit to 1 (clear the SEm0 bit, and stop the operation).**
**And after completion the receive operation, also clearing SWCm bit to 0 (SNOOZE mode release).**

**2. When SWCm = 1, BFFm1, OVFm1 flags are not operation.**

**Remarks 1.** <1> to <11> in the figure correspond to <1> to <11> in **Figure 12-71 Flowchart of SNOOZE Mode Operation (Once Startup)**.

**2.** m = 0; p = 00

**Figure 12-71. Flowchart of SNOOZE Mode Operation (Once Startup)**

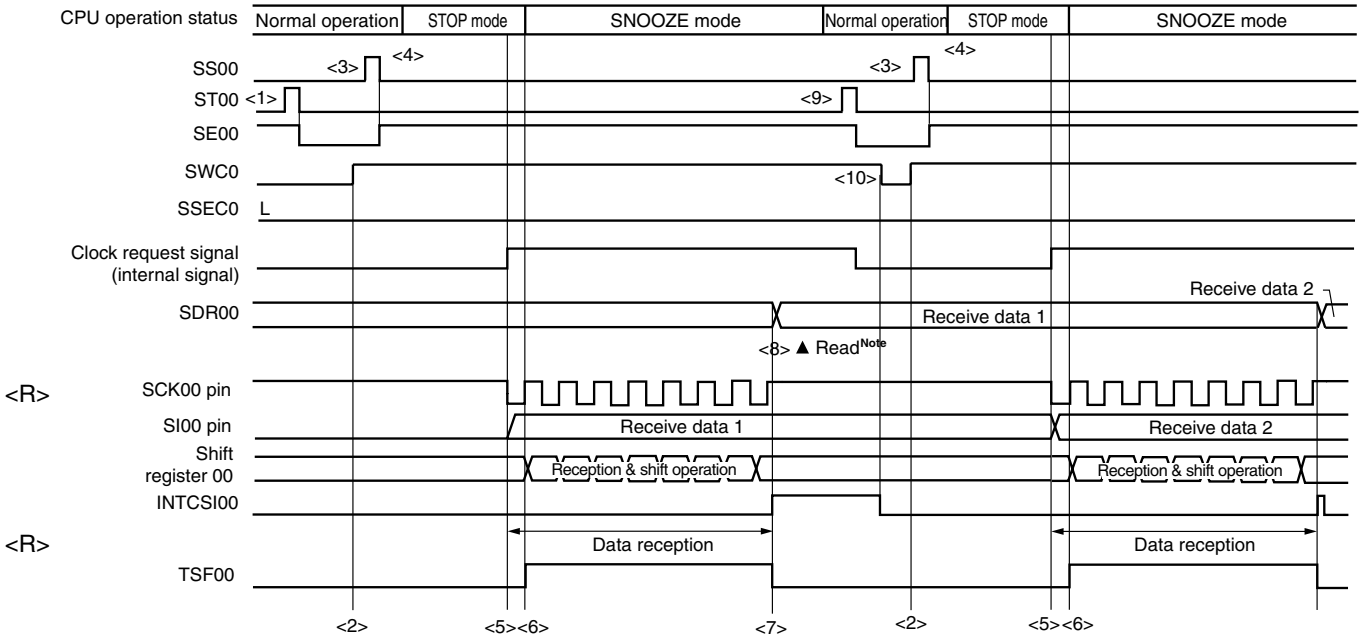



**Remarks 1.** <1> to <11> in the figure correspond to <1> to <11> in **Figure 12-70 Timing Chart of SNOOZE Mode Operation (Once Startup)**.

**2.** m = 0; p = 00

### (2) SNOOZE mode operation (continuous startup)

**Figure 12-72. Timing Chart of SNOOZE Mode Operation (Continuous Startup) (Type 1: DAPmn = 0, CKPmn = 0)**

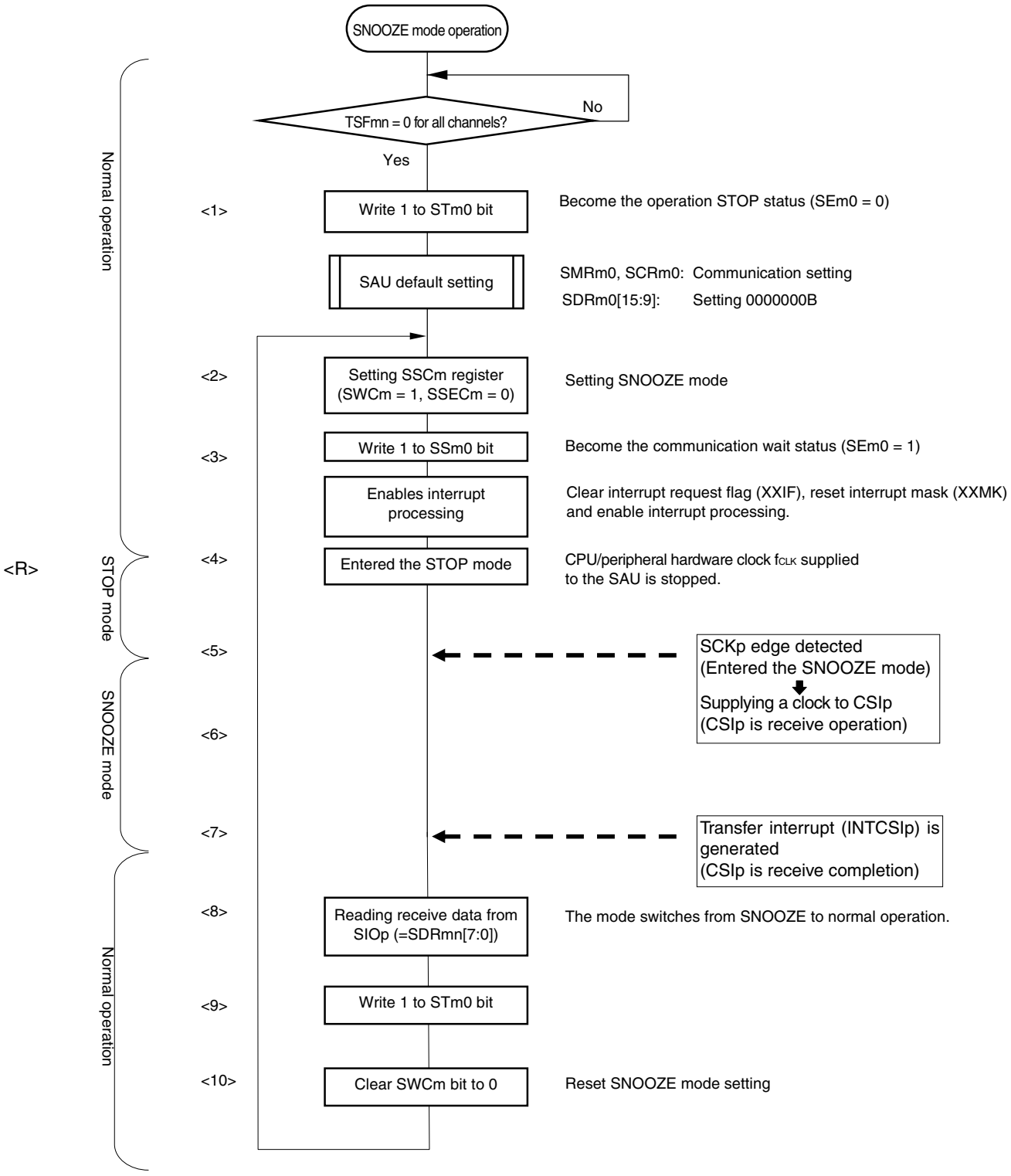



**Note** Only read received data while SWCm = 1 and before the next edge of the SCKp pin input is detected.

**Cautions 1. Before switching to the SNOOZE mode or after reception operation in the SNOOZE mode finishes, set the STm0 bit to 1 (clear the SEm0 bit, and stop the operation).**
**And after completion the receive operation, also clearing SWCm bit to 0 (SNOOZE release).**
**2. When SWCm = 1, BFFm1, OVFm1 flags are not operation.**

**Remarks 1.** <1> to <10> in the figure correspond to <1> to <10> in **Figure 12-73 Flowchart of SNOOZE Mode Operation (Continuous Startup)**.
**2.** m = 0; p = 00

**Figure 12-73. Flowchart of SNOOZE Mode Operation (Continuous Startup)**




**Remarks 1.** <1> to <10> in the figure correspond to <1> to <10> in **Figure 12-72 Timing Chart of SNOOZE Mode Operation (Continuous Startup)**.

**2.** m = 0; p = 00

### 12.5.8 Calculating transfer clock frequency

The transfer clock frequency for 3-wire serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) communication can be calculated by the following expressions.

**(1) Master**

(Transfer clock frequency) = {Operation clock ($f_{MCK}$) frequency of target channel} ÷ (SDRmn[15:9] + 1) ÷ 2 [Hz]

**(2) Slave**

(Transfer clock frequency) = {Frequency of serial clock (SCK) supplied by master}[Note] [Hz]

**Note** The permissible maximum transfer clock frequency is $f_{MCK}$/6.

**Remark** The value of SDRmn[15:9] is the value of bits 15 to 9 of serial data register mn (SDRmn) (0000000B to 1111111B) and therefore is 0 to 127.

The operation clock ($f_{MCK}$) is determined by serial clock select register m (SPSm) and bit 15 (CKSmn) of serial mode register mn (SMRmn).

**Table 12-2. Selection of Operation Clock for 3-Wire Serial I/O**

| SMRmn Register | SPSm Register | | | | | | | | Operation Clock ($f_{MCK}$)[Note] | |
|---|---|---|---|---|---|---|---|---|---|---|
| CKSmn | PRS m13 | PRS m12 | PRS m11 | PRS m10 | PRS m03 | PRS m02 | PRS m01 | PRS m00 | | $f_{CLK}$ = 32 MHz |
| 0 | X | X | X | X | 0 | 0 | 0 | 0 | $f_{CLK}$ | 32 MHz |
| | X | X | X | X | 0 | 0 | 0 | 1 | $f_{CLK}/2$ | 16 MHz |
| | X | X | X | X | 0 | 0 | 1 | 0 | $f_{CLK}/2^2$ | 8 MHz |
| | X | X | X | X | 0 | 0 | 1 | 1 | $f_{CLK}/2^3$ | 4 MHz |
| | X | X | X | X | 0 | 1 | 0 | 0 | $f_{CLK}/2^4$ | 2 MHz |
| | X | X | X | X | 0 | 1 | 0 | 1 | $f_{CLK}/2^5$ | 1 MHz |
| | X | X | X | X | 0 | 1 | 1 | 0 | $f_{CLK}/2^6$ | 500 kHz |
| | X | X | X | X | 0 | 1 | 1 | 1 | $f_{CLK}/2^7$ | 250 kHz |
| | X | X | X | X | 1 | 0 | 0 | 0 | $f_{CLK}/2^8$ | 125 kHz |
| | X | X | X | X | 1 | 0 | 0 | 1 | $f_{CLK}/2^9$ | 62.5 kHz |
| | X | X | X | X | 1 | 0 | 1 | 0 | $f_{CLK}/2^{10}$ | 31.25 kHz |
| | X | X | X | X | 1 | 0 | 1 | 1 | $f_{CLK}/2^{11}$ | 15.63 kHz |
| | X | X | X | X | 1 | 1 | 0 | 0 | $f_{CLK}/2^{12}$ | 7.81 kHz |
| | X | X | X | X | 1 | 1 | 0 | 1 | $f_{CLK}/2^{13}$ | 3.91 kHz |
| | X | X | X | X | 1 | 1 | 1 | 0 | $f_{CLK}/2^{14}$ | 1.95 kHz |
| | X | X | X | X | 1 | 1 | 1 | 1 | $f_{CLK}/2^{15}$ | 977 Hz |
| 1 | 0 | 0 | 0 | 0 | X | X | X | X | $f_{CLK}$ | 32 MHz |
| | 0 | 0 | 0 | 1 | X | X | X | X | $f_{CLK}/2$ | 16 MHz |
| | 0 | 0 | 1 | 0 | X | X | X | X | $f_{CLK}/2^2$ | 8 MHz |
| | 0 | 0 | 1 | 1 | X | X | X | X | $f_{CLK}/2^3$ | 4 MHz |
| | 0 | 1 | 0 | 0 | X | X | X | X | $f_{CLK}/2^4$ | 2 MHz |
| | 0 | 1 | 0 | 1 | X | X | X | X | $f_{CLK}/2^5$ | 1 MHz |
| | 0 | 1 | 1 | 0 | X | X | X | X | $f_{CLK}/2^6$ | 500 kHz |
| | 0 | 1 | 1 | 1 | X | X | X | X | $f_{CLK}/2^7$ | 250 kHz |
| | 1 | 0 | 0 | 0 | X | X | X | X | $f_{CLK}/2^8$ | 125 kHz |
| | 1 | 0 | 0 | 1 | X | X | X | X | $f_{CLK}/2^9$ | 62.5 kHz |
| | 1 | 0 | 1 | 0 | X | X | X | X | $f_{CLK}/2^{10}$ | 31.25 kHz |
| | 1 | 0 | 1 | 1 | X | X | X | X | $f_{CLK}/2^{11}$ | 15.63 kHz |
| | 1 | 1 | 0 | 0 | X | X | X | X | $f_{CLK}/2^{12}$ | 7.81 kHz |
| | 1 | 1 | 0 | 1 | X | X | X | X | $f_{CLK}/2^{13}$ | 3.91 kHz |
| | 1 | 1 | 1 | 0 | X | X | X | X | $f_{CLK}/2^{14}$ | 1.95 kHz |
| | 1 | 1 | 1 | 1 | X | X | X | X | $f_{CLK}/2^{15}$ | 977 Hz |
| Other than above | | | | | | | | | Setting prohibited | |

**Note** When changing the clock selected for $f_{CLK}$ (by changing the system clock control register (CKC) value), do so after having stopped (serial channel stop register m (STm) = 000FH) the operation of the serial array unit (SAU).

**Remarks 1.** X: Don't care

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

### 12.5.9 Procedure for processing errors that occurred during 3-wire serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) communication

The procedure for processing errors that occurred during 3-wire serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) communication is described in Figure 12-74.

**Figure 12-74. Processing Procedure in Case of Overrun Error**

| Software Manipulation | Hardware Status | Remark |
|---|---|---|
| Reads serial data register mn (SDRmn). → | The BFFmn bit of the SSRmn register is set to 0 and channel n is enabled to receive data. | This is to prevent an overrun error if the next reception is completed during error processing. |
| Reads serial status register mn (SSRmn). | | Error type is identified and the read value is used to clear error flag. |
| Writes 1 to serial flag clear trigger register mn (SIRmn). → | Error flag is cleared. | Error can be cleared only during reading, by writing the value read from the SSRmn register to the SIRmn register without modification. |

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

## 12.6 Operation of UART (UART0 to UART2) Communication

This is a start-stop synchronization function using two lines: serial data transmission (TxD) and serial data reception (RxD) lines. By using these two communication lines, each data frame, which consist of a start bit, data, parity bit, and stop bit, is transferred asynchronously (using the internal baud rate) between the microcontroller and the other communication party. Full-duplex UART communication can be performed by using a channel dedicated to transmission (even-numbered channel) and a channel dedicated to reception (odd-numbered channel). The LIN-bus can be implemented by using UART2, timer array unit o (channel 0) with an external interrupt (INTP0).

[Data transmission/reception]

- Data length of 7, 8, or 9 bits[Note]
- Select the MSB/LSB first
- Level setting of transmit/receive data (selecting whether to reverse the level)
- Parity bit appending and parity check functions
- Stop bit appending, stop bit check function

[Interrupt function]

- Transfer end interrupt/buffer empty interrupt
- Error interrupt in case of framing error, parity error, or overrun error

[Error detection flag]

- Framing error, parity error, or overrun error

In addition, UART0 reception (channel 1 of unit 0) supports the SNOOZE mode. When RxD0 pin input is detected while in the STOP mode, the SNOOZE mode makes data reception that does not require the CPU possible. Only following UART0 can be specified for the reception baud rate adjustment function.

The LIN-bus is accepted in UART2 (channels 0 and 1 of unit 1) (32, 48, and 64-pin products only).

<R> [LIN-bus functions]

- Wakeup signal detection
- Break field (BF) detection
- Sync field measurement, baud rate calculation

} Using the external interrupt (INTP0) and timer array unit 0 (channel 7)

**Note** Only following UART0 can be specified for the 9-bit data length.

UART0 uses channels 0 and 1 of SAU0.
UART1 uses channels 2 and 3 of SAU0.
UART2 uses channels 0 and 1 of SAU1.

- 25-pin products

| Unit | Channel | Used as CSI | Used as UART | Used as Simplified $I^2C$ |
|---|---|---|---|---|
| 0 | 0 | CSI00 | UART0 | IIC00 |
| | 1 | – | | – |
| | 2 | – | UART1 | – |
| | 3 | CSI11 | | IIC11 |

- 32-pin products

| Unit | Channel | Used as CSI | Used as UART | Used as Simplified $I^2C$ |
|---|---|---|---|---|
| 0 | 0 | CSI00 | UART0 | IIC00 |
| | 1 | – | | – |
| | 2 | – | UART1 | – |
| | 3 | CSI11 | | IIC11 |
| 1 | 0 | CSI20 | UART2 (supporting LIN-bus) | IIC20 |
| | 1 | – | | – |

- 48-pin products

| Unit | Channel | Used as CSI | Used as UART | Used as Simplified $I^2C$ |
|---|---|---|---|---|
| 0 | 0 | CSI00 | UART0 | IIC00 |
| | 1 | CSI01 | | IIC01 |
| | 2 | – | UART1 | – |
| | 3 | CSI11 | | IIC11 |
| 1 | 0 | CSI20 | UART2 (supporting LIN-bus) | IIC20 |
| | 1 | CSI21 | | IIC21 |

- 64-pin products

| Unit | Channel | Used as CSI | Used as UART | Used as Simplified $I^2C$ |
|---|---|---|---|---|
| 0 | 0 | CSI00 | UART0 | IIC00 |
| | 1 | CSI01 | | IIC01 |
| | 2 | CSI10 | UART1 | IIC10 |
| | 3 | CSI11 | | IIC11 |
| 1 | 0 | CSI20 | UART2 (supporting LIN-bus) | IIC20 |
| | 1 | CSI21 | | IIC21 |

Select any function for each channel. Only the selected function is possible. If UART0 is selected for channels 0 and 1 of unit 0, for example, these channels cannot be used for CSI00 and CSI01. At this time, however, channel 2, 3, or other channels of the same unit can be used for a function other than UART0, such as CSI10, UART1, and IIC10.

**Caution When using a serial array unit for UART, both the transmitter side (even-numbered channel) and the receiver side (odd-numbered channel) can only be used for UART.**

UART performs the following four types of communication operations.

- UART transmission (See **12.6.1**.)
- UART reception (See **12.6.2**.)
- LIN transmission (UART2 only) (See **12.7.1**.)
- LIN reception (UART2 only) (See **12.7.2**.)

### 12.6.1 UART transmission

<R> UART transmission is an operation to transmit data from the RL78 microcontroller to another device asynchronously (start-stop synchronization).

Of two channels used for UART, the even channel is used for UART transmission.

| UART | UART0 | UART1 | UART2 |
|---|---|---|---|
| Target channel | Channel 0 of SAU0 | Channel 2 of SAU0 | Channel 0 of SAU1 |
| Pins used | TxD0 | TxD1 | TxD2 |
| Interrupt | INTST0 | INTST1 | INTST2 |
| | Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected. | | |
| Error detection flag | None | | |
| Transfer data length | 7, 8, or 9 bits[Note 1] | | |
| Transfer rate[Note 2] | Max. $f_{MCK}$/6 [bps] (SDRmn[15:9] = 2 or more), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps] | | |
| Data phase | Non-reverse output (default: high level)<br>Reverse output (default: low level) | | |
| Parity bit | The following selectable<br>• No parity bit<br>• Appending 0 parity<br>• Appending even parity<br>• Appending odd parity | | |
| Stop bit | The following selectable<br>• Appending 1 bit<br>• Appending 2 bits | | |
| Data direction | MSB or LSB first | | |

**Notes 1.** Only following UART0 can be specified for the 9-bit data length.

**2.** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C), CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to +105°C)**).

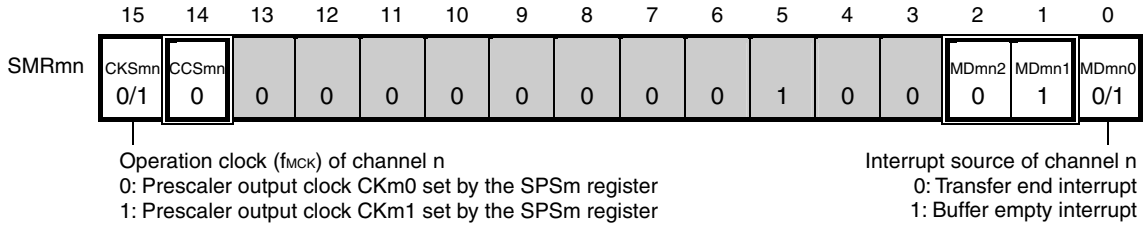
**Remarks 1.** $f_{MCK}$: Operation clock frequency of target channel
$f_{CLK}$: System clock frequency

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), mn = 00, 02, 10

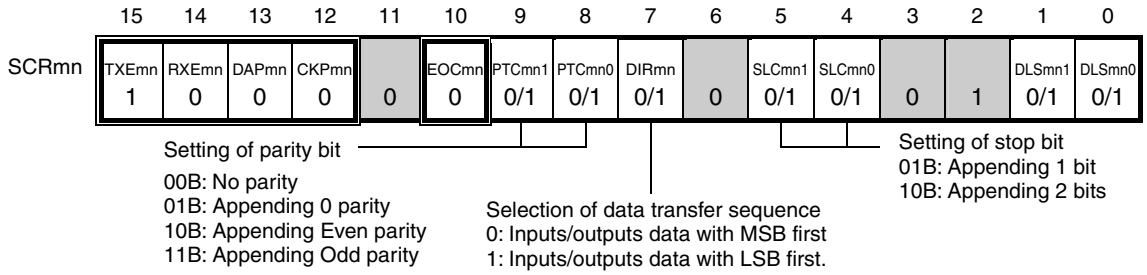
### (1) Register setting

**Figure 12-75. Example of Contents of Registers for UART Transmission of UART (UART0 to UART2) (1/2)**

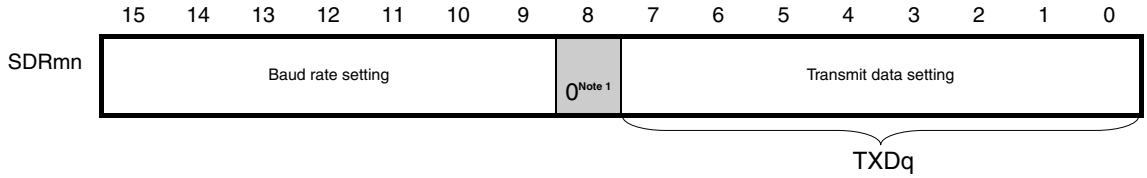
**(a) Serial mode register mn (SMRmn)**



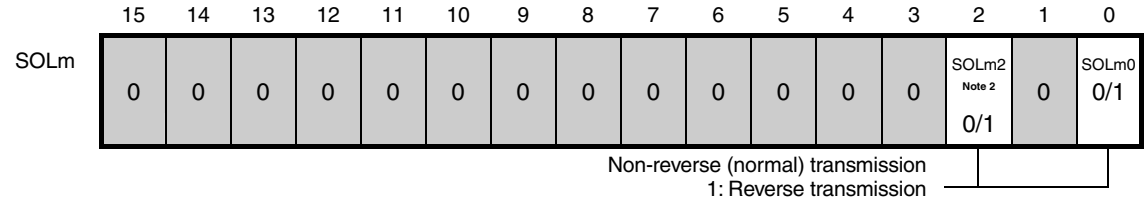

**(b) Serial communication operation setting register mn (SCRmn)**




**(c) Serial data register mn (SDRmn) (lower 8 bits: TXDq)**




**(d) Serial output level register m (SOLm) … Sets only the bits of the target channel.**




<R> **Notes 1.** SCR00 only. This bit is fixed to 1 for the other registers.

**2.** When UART0 performs 9-bit communication (by setting the DLS001 and DLS000 bits of the SCR00 register to 1), bits 0 to 8 of the SDR00 register are used as the transmission data specification area. Only following UART0 can be specified for the 9-bit data length.

**3.** Unit 0 only

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), q: UART number (q = 0 to 2), mn = 00, 02, 10

**2.** □: Setting is fixed in the UART transmission mode, ▒: Setting disabled (set to the initial value)
×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)
0/1: Set to 0 or 1 depending on the usage of the user

**Figure 12-75. Example of Contents of Registers for UART Transmission of UART (UART0 to UART2) (2/2)**

**(e) Serial output register m (SOm) … Sets only the bits of the target channel.**

SOm

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | CKOm3 Note 1 × | CKOm2 Note 1 × | CKOm1 × | CKOm0 × | 0 | 0 | 0 | 0 | Som3 Note 1 × | SOm2 Notes 1,2 0/1 | SOm1 × | SOm0 Note 2 0/1 |

0: Serial data output value is “0”
1: Serial data output value is “1”

**(f) Serial output enable register m (SOEm) … Sets only the bits of the target channel to 1.**

SOEm

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SOEm3 Note 1 × | SOEm2 Note 1 0/1 | SOEm1 × | SOEm0 0/1 |

**(g) Serial channel start register m (SSm) … Sets only the bits of the target channel to 1.**

SSm

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SSm3 Note 1 × | SSm2 Note 1 0/1 | SSm1 × | SSm0 0/1 |

**Notes 1.** Before transmission is started, be sure to set to 1 when the SOLmn bit of the target channel is set to 0, and set to 0 when the SOLmn bit of the target channel is set to 1. The value varies depending on the communication data during communication operation.

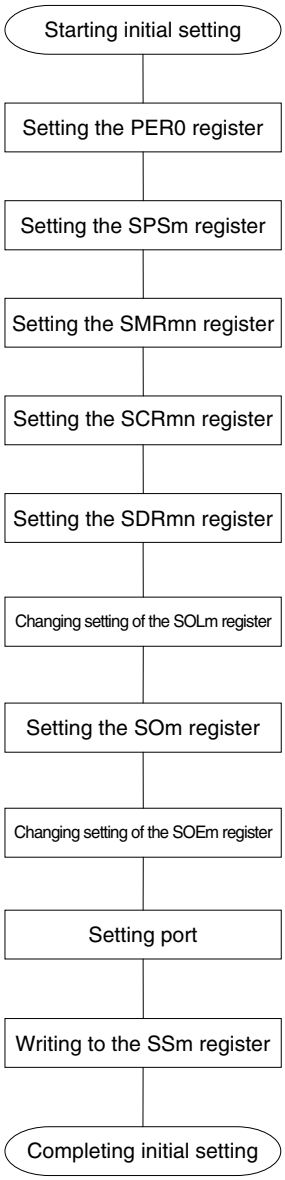
**2.** Unit 0 only

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), mn = 00, 02, 10

**2.** ☐ : Setting disabled (set to the initial value)
×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)
0/1: Set to 0 or 1 depending on the usage of the user

### (2) Operation procedure

**Figure 12-76. Initial Setting Procedure for UART Transmission**




Release the serial array unit from the reset status and start clock supply.

Set the operation clock.

Set an operation mode, etc.

Set a communication format.

Set a transfer baud rate (setting the transfer clock by dividing the operation clock ($f_{MCK}$)).

Set an output data level.

Set the initial output level of the serial data (SOmn).

Set the SOEmn bit to 1 and enable data output of the target channel.

Enable data output of the target channel by setting a port register and a port mode register.

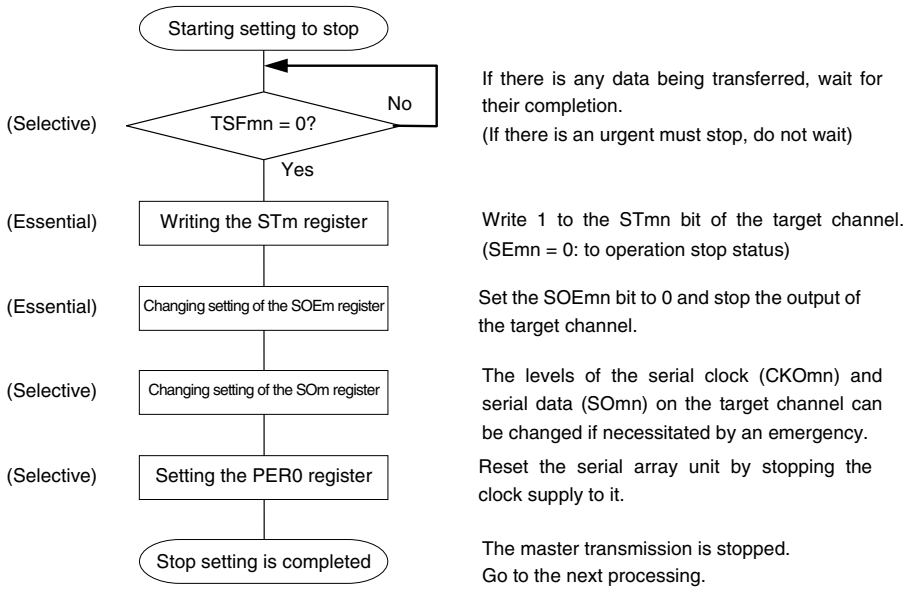
Set the SSmn bit of the target channel to 1 and set the SEmn bit to 1 (to enable operation).

<R> Initial setting is completed.
Set transmit data to the SDRmn[7:0] bits (TXDq register) (8 bits) or the SDRmn[8:0] bits (9 bits) and start communication.

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), q: UART number (q = 0 to 2)
mn = 00, 02, 10

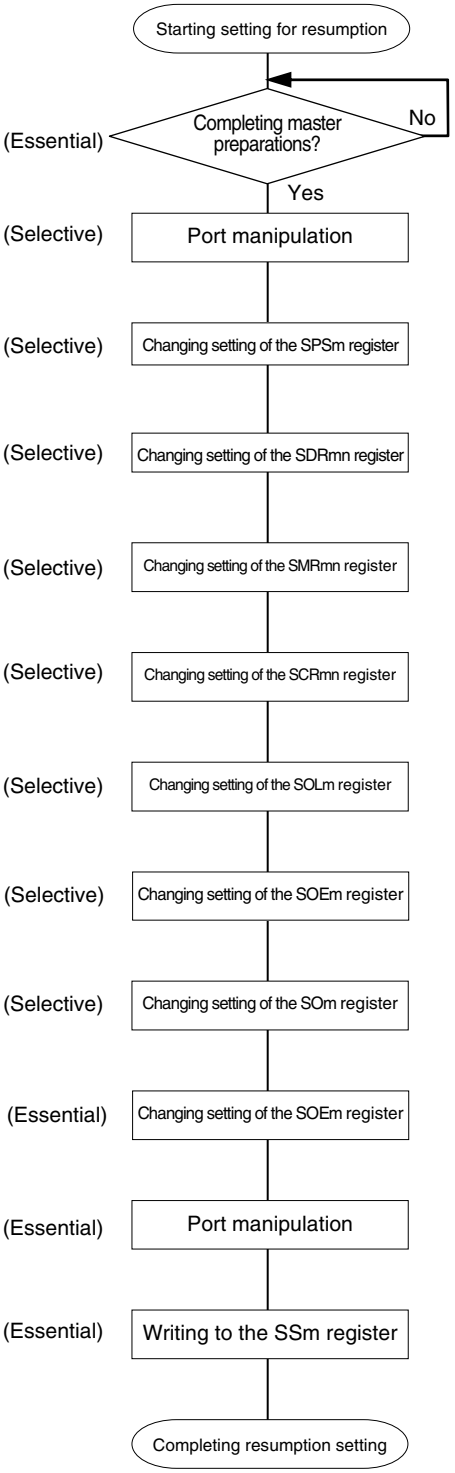
**Figure 12-77. Procedure for Stopping UART Transmission**

<R>




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), mn = 00, 02, 10

**Figure 12-78. Procedure for Resuming UART Transmission**




Wait until stop the communication target or communication operation completed

Disable data output of the target channel by setting a port register and a port mode register.

Re-set the register to change the operation clock setting.

Re-set the register to change the transfer baud rate setting (setting the transfer clock by dividing the operation clock ($f_{MCK}$)).

Re-set the register to change serial mode register mn (SMRmn) setting.

Re-set the register to change the serial communication operation setting register mn (SCRmn) setting.

Re-set the register to change serial output level register m (SOLm) setting.

Clear the SOEmn bit to 0 and stop output.

Set the initial output level of the serial data (SOmn).

Set the SOEmn bit to 1 and enable output.

Enable data output of the target channel by setting a port register and a port mode register.

Set the SSmn bit of the target channel to 1 and set the SEmn bit to 1 (to enable operation).

<R> Setting is completed
Set transmit data to the SDRmn[7:0] bits (TXDq register) (8 bits) or the SDRmn[8:0] bits (9 bits) and start communication.

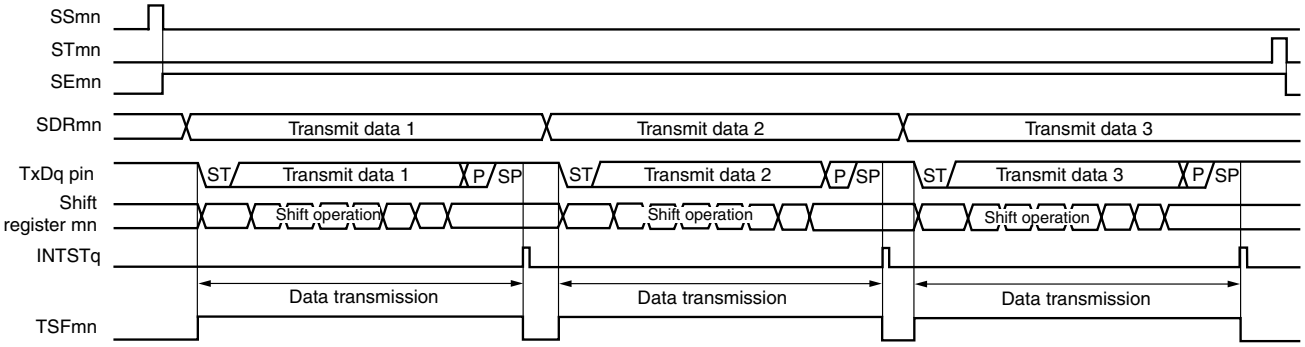
**Remarks 1.** If PER0 is rewritten while stopping the master transmission and the clock supply is stopped, wait until the transmission target stops or transmission finishes, and then perform initialization instead of restarting the transmission.

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), q: UART number (q = 0 to 2), mn = 00, 02, 10

### (3) Processing flow (in single-transmission mode)

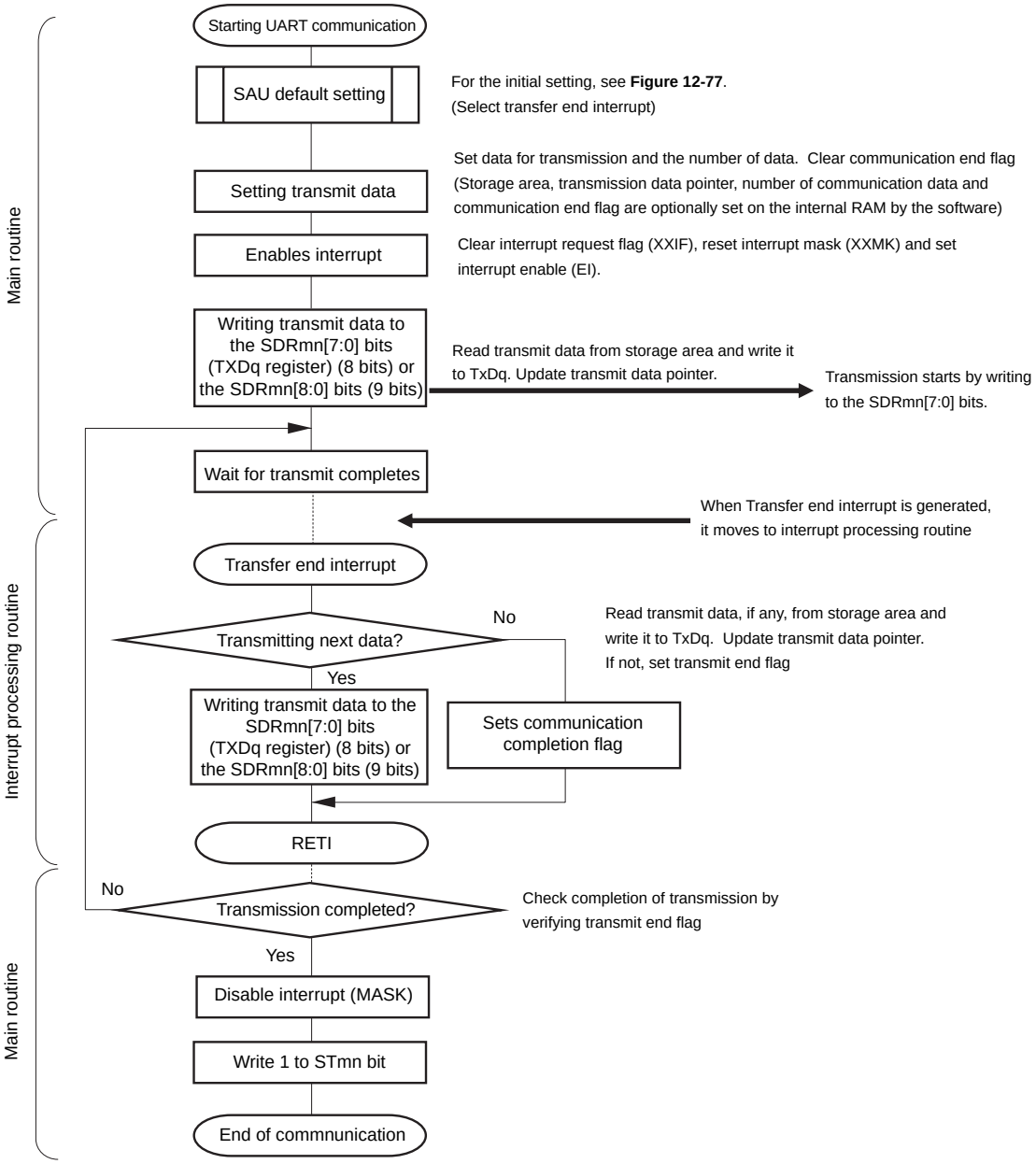
**Figure 12-79. Timing Chart of UART Transmission (in Single-Transmission Mode)**

<R>




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), q: UART number (q = 0 to 2)
mn = 00, 02, 10

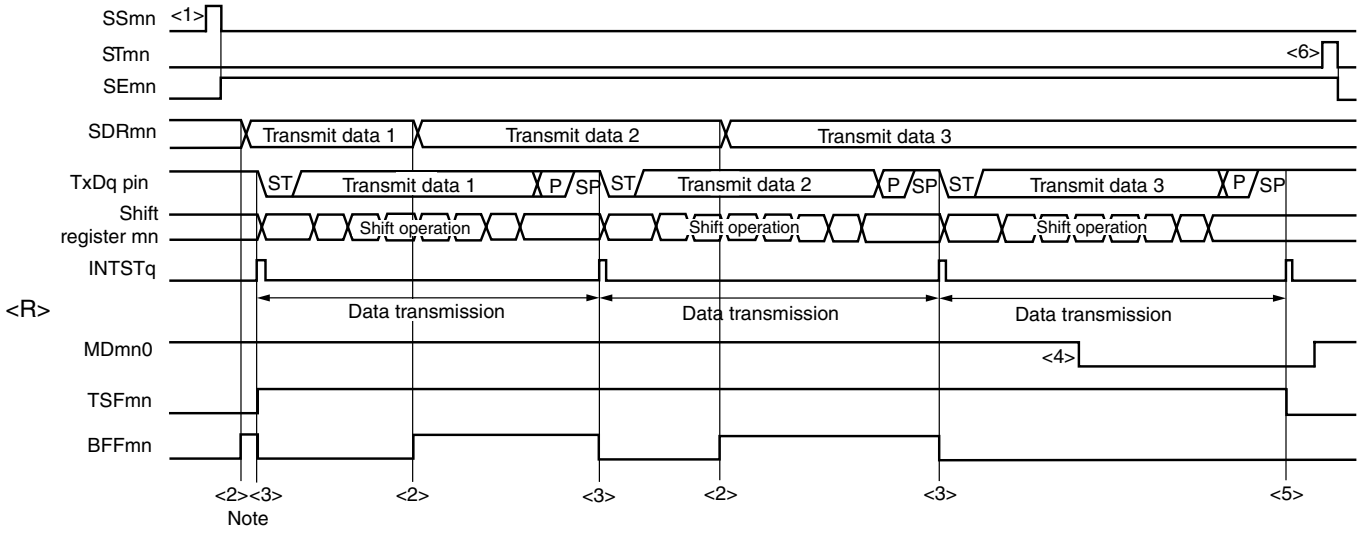
**Figure 12-80. Flowchart of UART Transmission (in Single-Transmission Mode)**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), q: UART number (q = 0 to 2)
mn = 00, 02, 10

### (4) Processing flow (in continuous transmission mode)

**Figure 12-81. Timing Chart of UART Transmission (in Continuous Transmission Mode)**

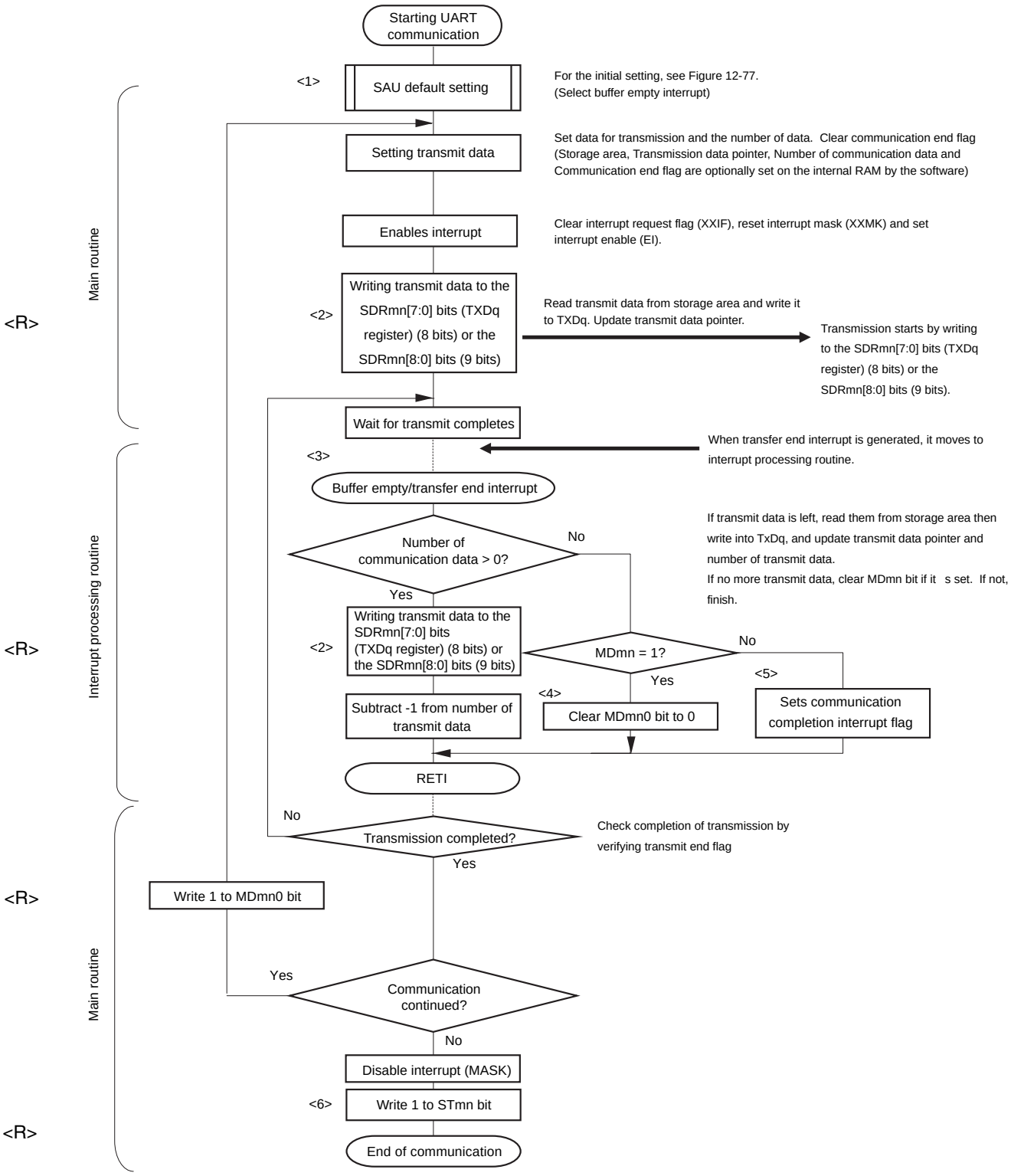



**Note** If transmit data is written to the SDRmn register while the BFFmn bit of serial status register mn (SSRmn) is 1 (valid data is stored in serial data register mn (SDRmn)), the transmit data is overwritten.

**Caution The MDmn0 bit of serial mode register mn (SSRmn) can be rewritten even during operation. However, rewrite it before transfer of the last bit is started, so that it will be rewritten before the transfer end interrupt of the last transmit data.**

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), q: UART number (q = 0 to 2)
mn = 00, 02, 10

**Figure 12-82. Flowchart of UART Transmission (in Continuous Transmission Mode)**




**Remarks 1.** <1> to <6> in the figure correspond to <1> to <6> in **Figure 12-81 Timing Chart of UART Transmission (in Continuous Transmission Mode)**.

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), q: UART number (q = 0 to 2)
mn = 00, 02, 10

### 12.6.2 UART reception

<R> UART reception is an operation wherein the RL78 microcontroller asynchronously receives data from another device (start-stop synchronization).

For UART reception, the odd-number channel of the two channels used for UART is used. The SMR register of both the odd- and even-numbered channels must be set.

| UART | UART0 | UART1 | UART2 |
|---|---|---|---|
| Target channel | Channel 1 of SAU0 | Channel 3 of SAU0 | Channel 1 of SAU1 |
| Pins used | RxD0 | RxD1 | RxD2 |
| Interrupt | INTSR0 | INTSR1 | INTSR2 |
| | Transfer end interrupt only (Setting the buffer empty interrupt is prohibited.) | | |
| Error interrupt | INTSRE0 | INTSRE1 | INTSRE2 |
| Error detection flag | • Framing error detection flag (FEFmn)<br>• Parity error detection flag (PEFmn)<br>• Overrun error detection flag (OVFmn) | | |
| Transfer data length | 7, 8 or 9 bits Note 1 | | |
| Transfer rate Note 2 | Max. $f_{MCK}/6$ [bps] (SDRmn[15:9] = 2 or more), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps] | | |
| Data phase | Non-reverse output (default: high level)<br>Reverse output (default: low level) | | |
| Parity bit | The following selectable<br>• No parity bit (no parity check)<br>• No parity judgment (0 parity)<br>• Appending even parity<br>• Appending odd parity | | |
| Stop bit | 1 bit addition | | |
| Data direction | MSB or LSB first | | |

**Notes 1.** Only following UART0 can be specified for the 8-bit data length.

**2.** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = −40 to +85°C), CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = −40 to +105°C)**).

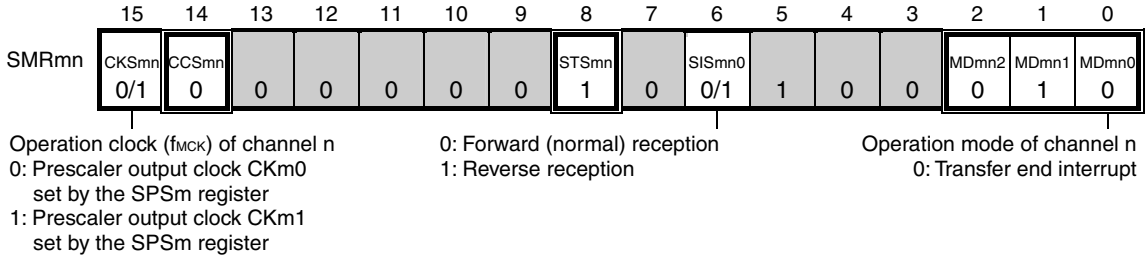
**Remarks 1.** $f_{MCK}$: Operation clock frequency of target channel
$f_{CLK}$: System clock frequency

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 1, 3), mn = 01, 03, 11

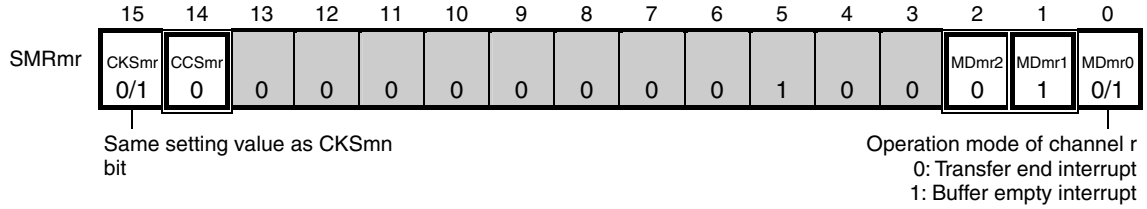
### (1) Register setting

**Figure 12-83. Example of Contents of Registers for UART Reception of UART (UART0 to UART2) (1/2)**

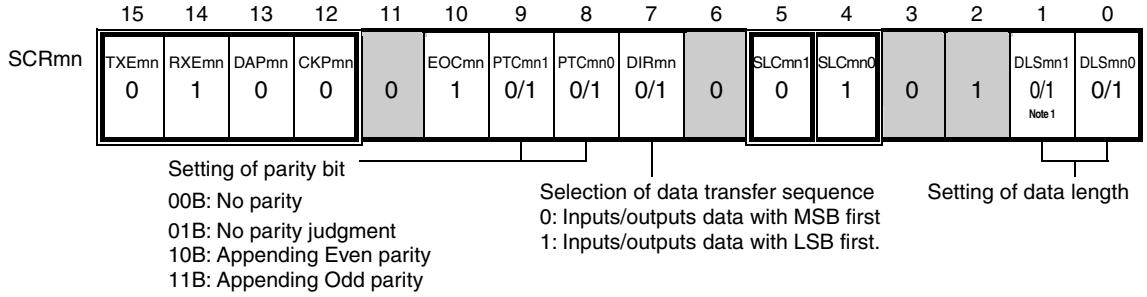
**(a) Serial mode register mn (SMRmn)**



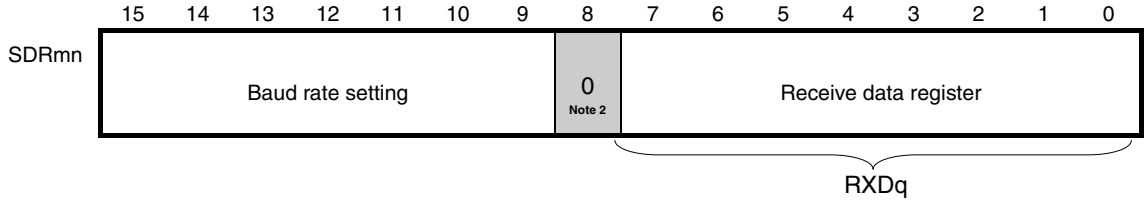

**(b) Serial mode register mr (SMRmr)**




**(c) Serial communication operation setting register mn (SCRmn)**




**(d) Serial data register mn (SDRmn) (lower 8 bits: RXDq)**




<R> **Notes 1.** SCR01 only. This bit is fixed to 1 for the other registers.

**2.** When UART performs 9-bit communication, bits 0 to 8 of the SDRm1 register are used as the transmission data specification area. Only following UART0 can be specified for the 8-bit data length.

**Caution For the UART reception, be sure to set the SMRmr register of channel r to UART transmission mode that is to be paired with channel n.**

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 1, 3), mn = 01, 03, 11
r: Channel number (r = n – 1), q: UART number (q = 0 to 2)

**2.** ☐: Setting is fixed in the UART reception mode, ▒: Setting disabled (set to the initial value)
0/1: Set to 0 or 1 depending on the usage of the user

**Figure 12-83. Example of Contents of Registers for UART Reception of UART (UART0 to UART2) (2/2)**

**(e) Serial output register m (SOm) … The register that not used in this mode.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SOm | 0 | 0 | 0 | 0 | CKOm3 Note × | CKOm2 Note × | CKOm1 × | CKOm0 × | 0 | 0 | 0 | 0 | SOm3 Note × | SOm2 Note × | SOm1 × | SOm0 × |

**(f) Serial output enable register m (SOEm) …The register that not used in this mode.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SOEm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SOEm3 Note × | SOEm2 Note × | SOEm1 × | SOEm0 × |

**(g) Serial channel start register m (SSm) … Sets only the bits of the target channel is 1.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SSm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SSm3 Note 0/1 | SSm2 Note × | SSm1 0/1 | SSm0 × |

**Note** Unit 0 only

**Remarks 1.** m: Unit number (m = 0, 1)

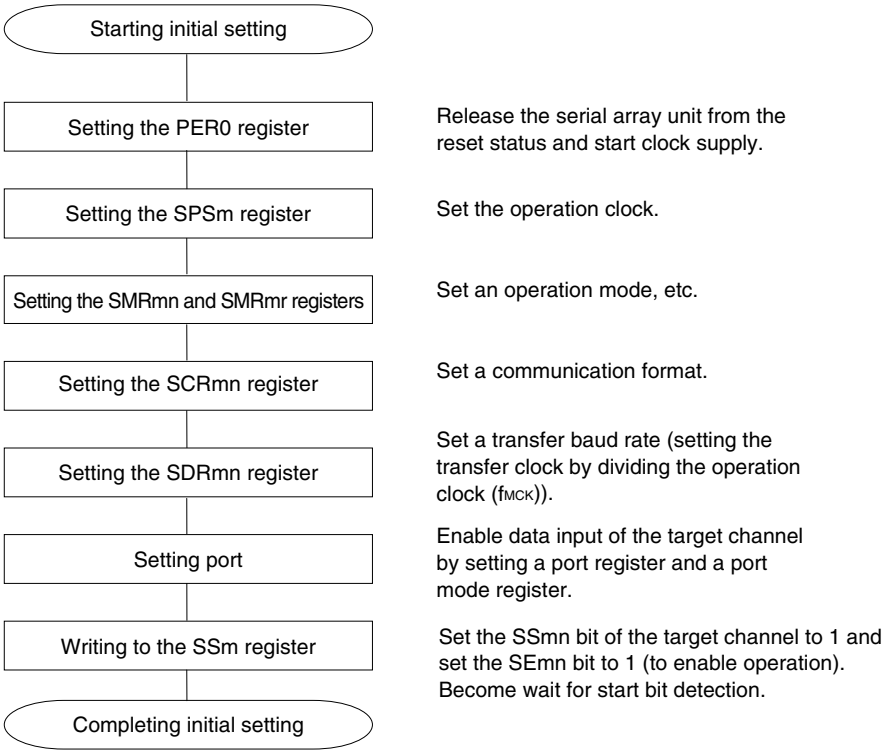
**2.** ☐ : Setting disabled (set to the initial value)

×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)

0/1: Set to 0 or 1 depending on the usage of the user

### (2) Operation procedure

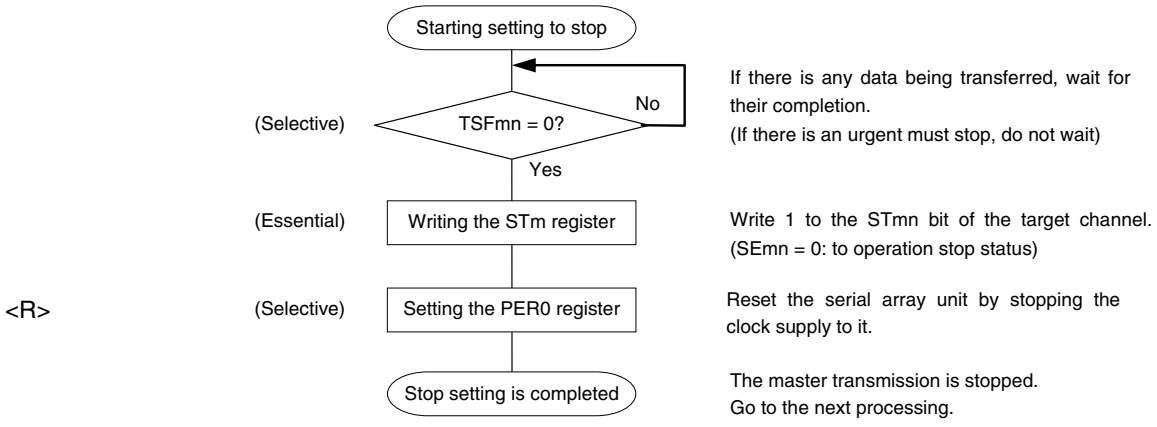
**Figure 12-84. Initial Setting Procedure for UART Reception**




**Caution Set the RXEmn bit of SCRmn register to 1, and then be sure to set SSmn to 1 after 4 or more $f_{MCK}$ clocks have elapsed.**

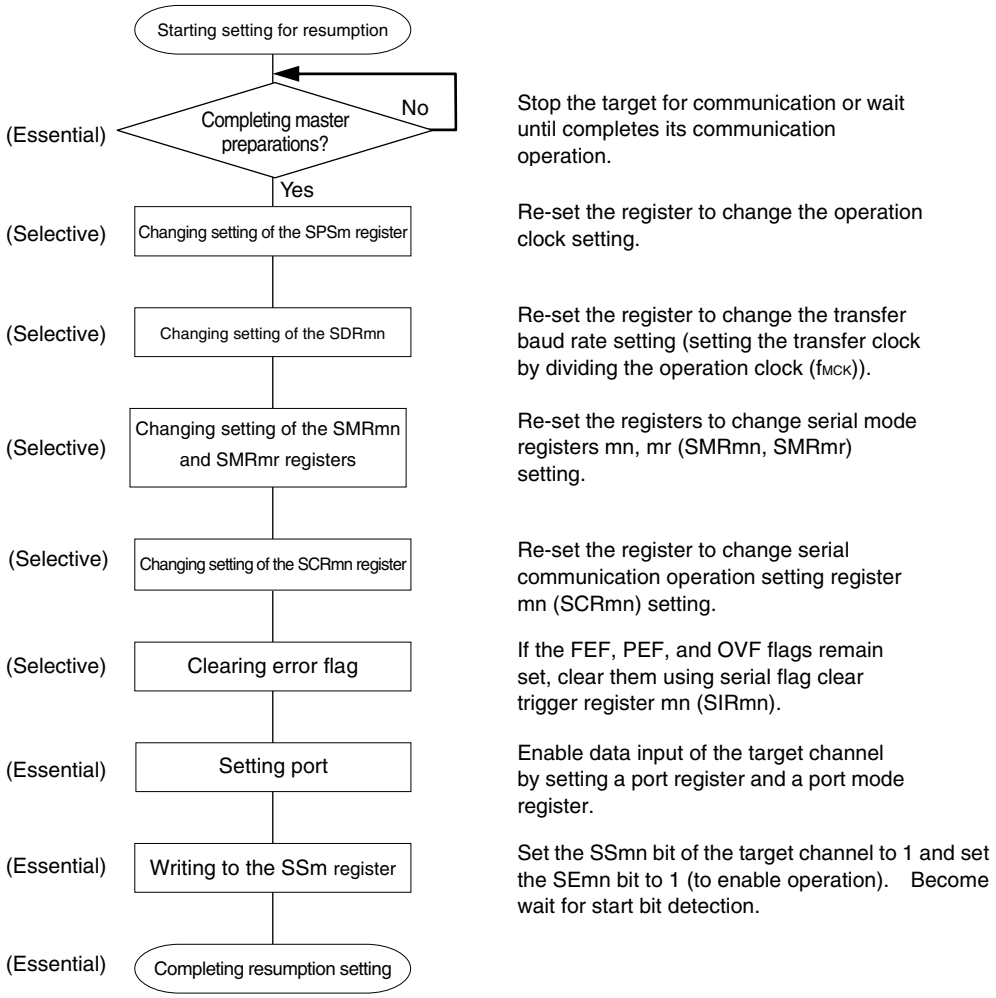
**Figure 12-85. Procedure for Stopping UART Reception**

<R>




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 1, 3), mn = 01, 03, 11
r: Channel number (r = n − 1)

**Figure 12-86. Procedure for Resuming UART Reception**



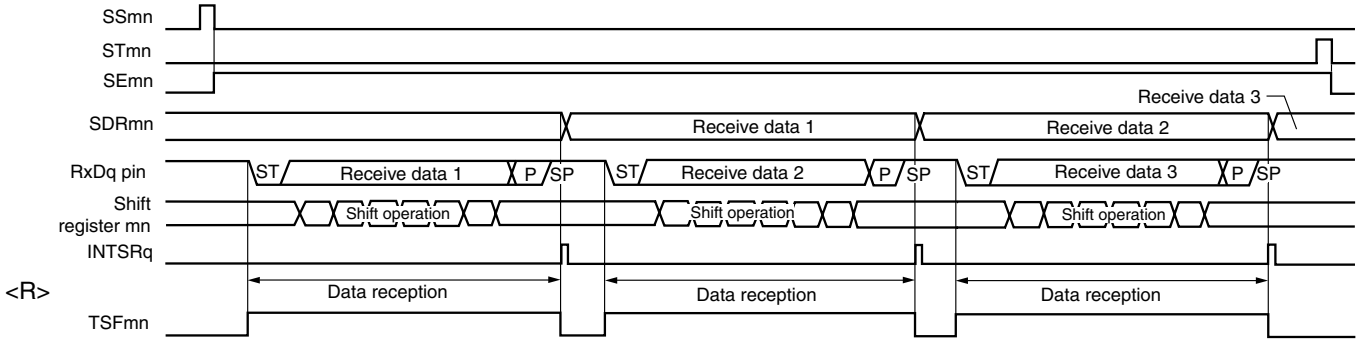

**Caution After is set RXEmn bit to 1 of SCRmn register, set the SSmn = 1 from an interval of at least four clocks of $f_{MCK}$.**

**Remarks 1.** If PER0 is rewritten while stopping the master transmission and the clock supply is stopped, wait until the transmission target (slave) stops or transmission finishes, and then perform initialization instead of restarting the transmission.

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 1, 3), mn = 01, 03, 11
r: Channel number (r = n − 1)

### (3) Processing flow

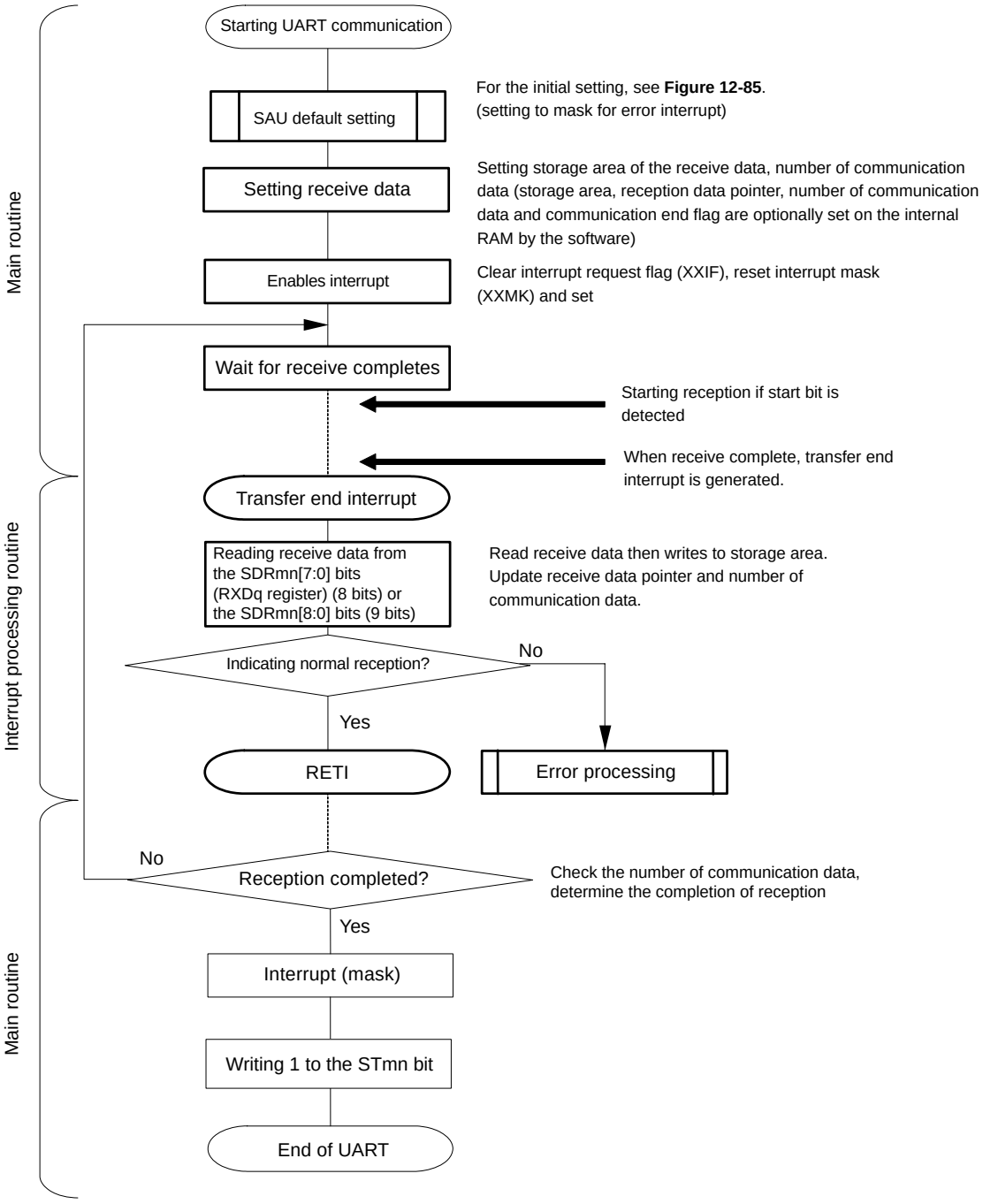
**Figure 12-87. Timing Chart of UART Reception**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 1, 3), mn = 01, 03, 11
r: Channel number (r = n − 1), q: UART number (q = 0 to 2)

**Figure 12-88. Flowchart of UART Reception**

<R>




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 1, 3), mn = 01, 03, 11
r: Channel number (r = n − 1), q: UART number (q = 0 to 2)

### 12.6.3 SNOOZE mode function

<R> The SNOOZE mode makes the UART perform reception operations upon RxDq pin input detection while in the STOP mode. Normally the UART stops communication in the STOP mode. However, using the SNOOZE mode enables the UART to perform reception operations without CPU operation. Only UART0 channel can be set to the SNOOZE mode.

When using UARTq in the SNOOZE mode, perform the following steps before entering the STOP mode. (See **Figure 12-92 Flowchart of SNOOZE Mode Operation (EOCm1 = 0, SSECm = 0/1 or EOCm1 = 1, SSECm = 0)** and **Figure 12-94 Flowchart of SNOOZE Mode Operation (EOCm1 = 1, SSECm = 1)**.)

- In the SNOOZE mode, the baud rate setting for UART reception needs to be changed to a value different from that in normal operation. Set the SPSm register and bits 15 to 9 of the SDRmn register with reference to Table 12-3.
- Set the EOCmn and SSECmn bits. This is for enabling or stopping generation of an error interrupt (INTSRE0) when a communication error occurs.
- Set the SWCm bit of serial standby control register m (SSCm) to 1 immediately before entering the STOP mode. After initial setup finishes, set the SSm1 bit of serial channel start register m (SSm) to 1, which sets the SEm1 bit to 1.

If edge of RxDq is detection after switching to the STOP mode, the UART reception is started.

**Cautions 1. The SNOOZE mode can only be specified when the high-speed on-chip oscillator clock ($f_{IH}$) is selected for $f_{CLK}$.**

**2. The maximum transfer rate when using UARTq in the SNOOZE mode is 4800 bps.**

**3. When SWCm = 1, UARTq can be used only when the reception operation is started in the STOP mode. When used simultaneously with another SNOOZE mode function or interrupt, if the reception operation is started in a state other than the STOP mode, such as those given below, data may not be received correctly and a framing error or parity error may be generated.**

- **When after the SWCm bit has been set to 1, the reception operation is started before the STOP mode is entered**
- **When the reception operation is started while another function is in the SNOOZE mode**
- **When after returning from the STOP mode to normal operation due to an interrupt or other cause, the reception operation is started before the SWCm bit is returned to 0**

**4. If a parity error, framing error, or overrun error occurs while the SSECm bit is set to 1, the PEFmn, FEFmn, or OVFmn flag is not set and an error interrupt (INTSREq) is not generated. Therefore, when the setting of SSECm = 1 is made, clear the PEFmn, FEFmn, or OVFmn flag before setting the SWC0 bit to 1 and read the value in bits 7 to 0 (RxDq register) of the SDRm1 register.**

**Remark** m = 0; q = 0

**Table 12-3. UART Reception Baud Rate Setting in the SNOOZE Mode**

| High-speed On-chip Oscillator ($f_{IH}$) | UART Reception Baud Rate Setting in the SNOOZE Mode | | | |
|---|---|---|---|---|
| | Baud Rate 4800 bps | | | |
| | Operation Cock ($f_{MCK}$) | SDRmn [15:9] | Maximum Permissible Value | Minimum Permissible Value |
| 32 MHz ± 1.0%Note | $f_{CLK}/2^5$ | 105 | 2.27% | –1.53% |
| 24 MHz ± 1.0%Note | $f_{CLK}/2^5$ | 79 | 1.60% | –2.18% |
| 16 MHz ± 1.0%Note | $f_{CLK}/2^4$ | 105 | 2.27% | –1.53% |
| 12 MHz ± 1.0%Note | $f_{CLK}/2^4$ | 79 | 1.60% | –2.19% |
| 8 MHz ± 1.0%Note | $f_{CLK}/2^3$ | 105 | 2.27% | –1.53% |
| 6 MHz ± 1.0%Note | $f_{CLK}/2^3$ | 79 | 1.60% | –2.19% |
| 4 MHz ± 1.0%Note | $f_{CLK}/2^2$ | 105 | 2.27% | –1.53% |
| 3 MHz ± 1.0%Note | $f_{CLK}/2^2$ | 79 | 1.60% | –2.19% |
| 2 MHz ± 1.0%Note | $f_{CLK}/2$ | 105 | 2.27% | –1.54% |
| 1 MHz ± 1.0%Note | $f_{CLK}$ | 105 | 2.27% | –1.57% |

**Note** The allowable range is narrowed as described below when the accuracy of the high-speed on-chip oscillator clock frequency is ±1.5% or ±2.0%.

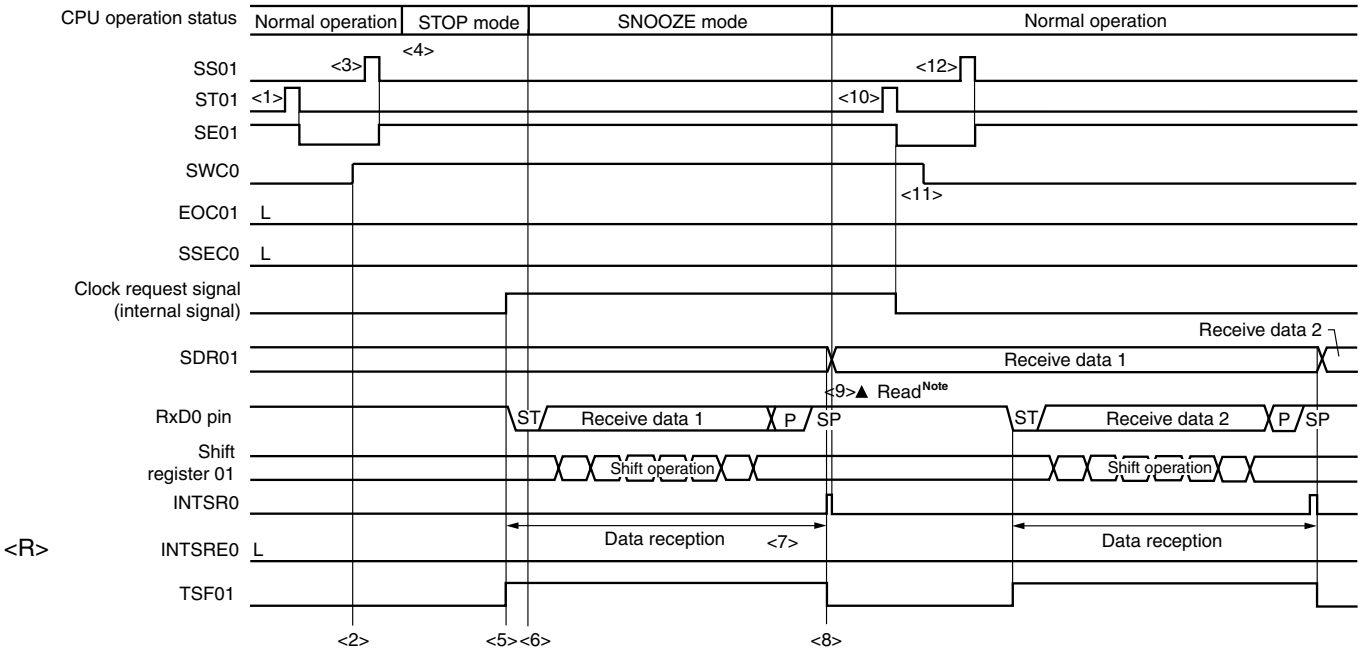
- In the case of $f_{IH}$ ± 1.5%, perform (Maximum permissible value – 0.5%) and (Minimum permissible value + 0.5%) to the values in the above table.
- In the case of $f_{IH}$ ± 2.0%, perform (Maximum permissible value – 1.0%) and (Minimum permissible value + 1.0%) to the values in the above table.

**Remark** The maximum and minimum allowable values indicate the allowable baud rate values for UART reception.
Specify the baud rate so that the baud rate on the transmission side is within this range.

### (1) SNOOZE mode operation (EOCm1 = 0, SSECm = 0/1)

Because EOCm1 = 0, an error interrupt (INTSRE0) is not generated even if a communication error occurs independently of the setting of SSECmn bit. Transfer end interrupt (INTSR0) is generated.

**Figure 12-89. Timing Chart of SNOOZE Mode Operation (OCm1 = 0, SSECm = 0/1)**




**Note** Read the received data when SWCm is 1

**Caution Before switching to the SNOOZE mode or after reception operation in the SNOOZE mode finishes, set the STm1 bit to 1 (clear the SEm1 bit, and stop the operation).**
**And after completion the receive operation, also clearing SWCm bit to 0 (SNOOZE mode release).**

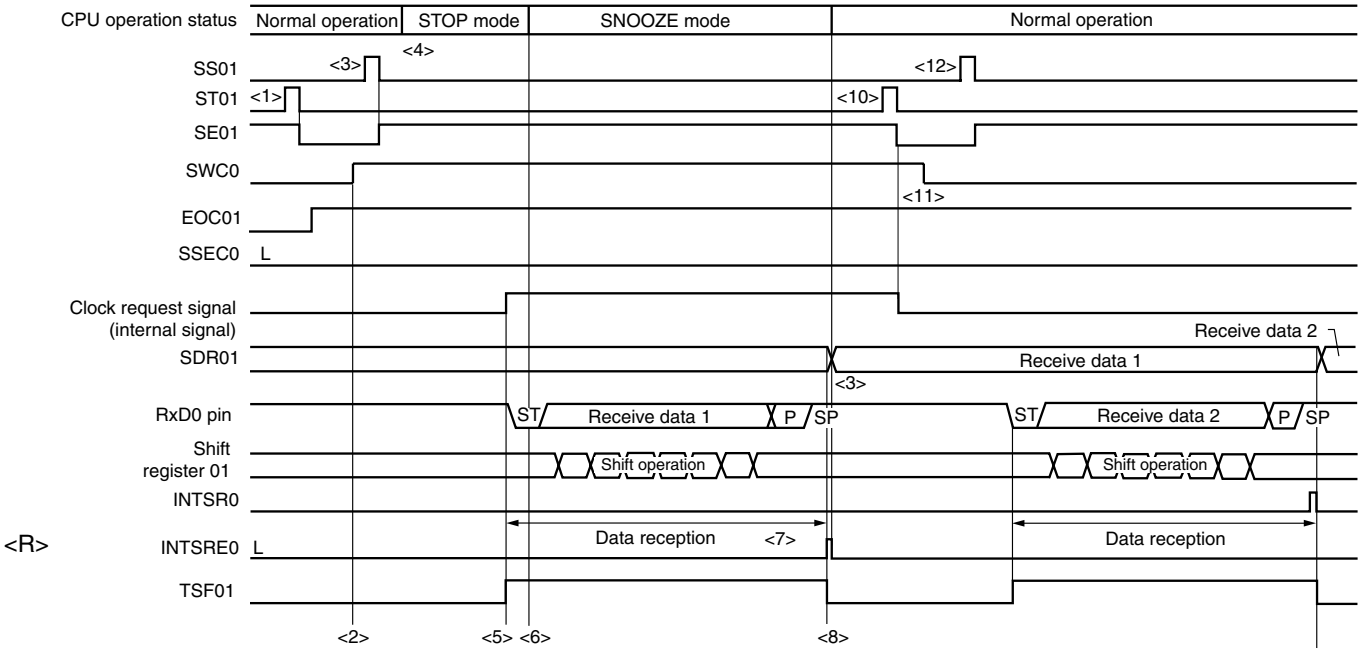
**Remarks 1.** <1> to <11> in the figure correspond to <1> to <11> in **Figure 12-91 Flowchart of SNOOZE Mode Operation (EOCm1 = 0, SSECm = 0/1 or EOCm1 = 1, SSECm = 0)**.

**2.** m = 0; q = 0

**(2) SNOOZE mode operation (EOCm1 = 1, SSECm = 0: error interrupt (INTSRE0) generation enable)**

Because EOCm1 = 1, SSECm = 0, an error interrupt (INTSREq) is generated when a communication error occurs.

**Figure 12-90. Timing Chart of SNOOZE Mode Operation (EOCm1 = 1, SSECm = 0)**

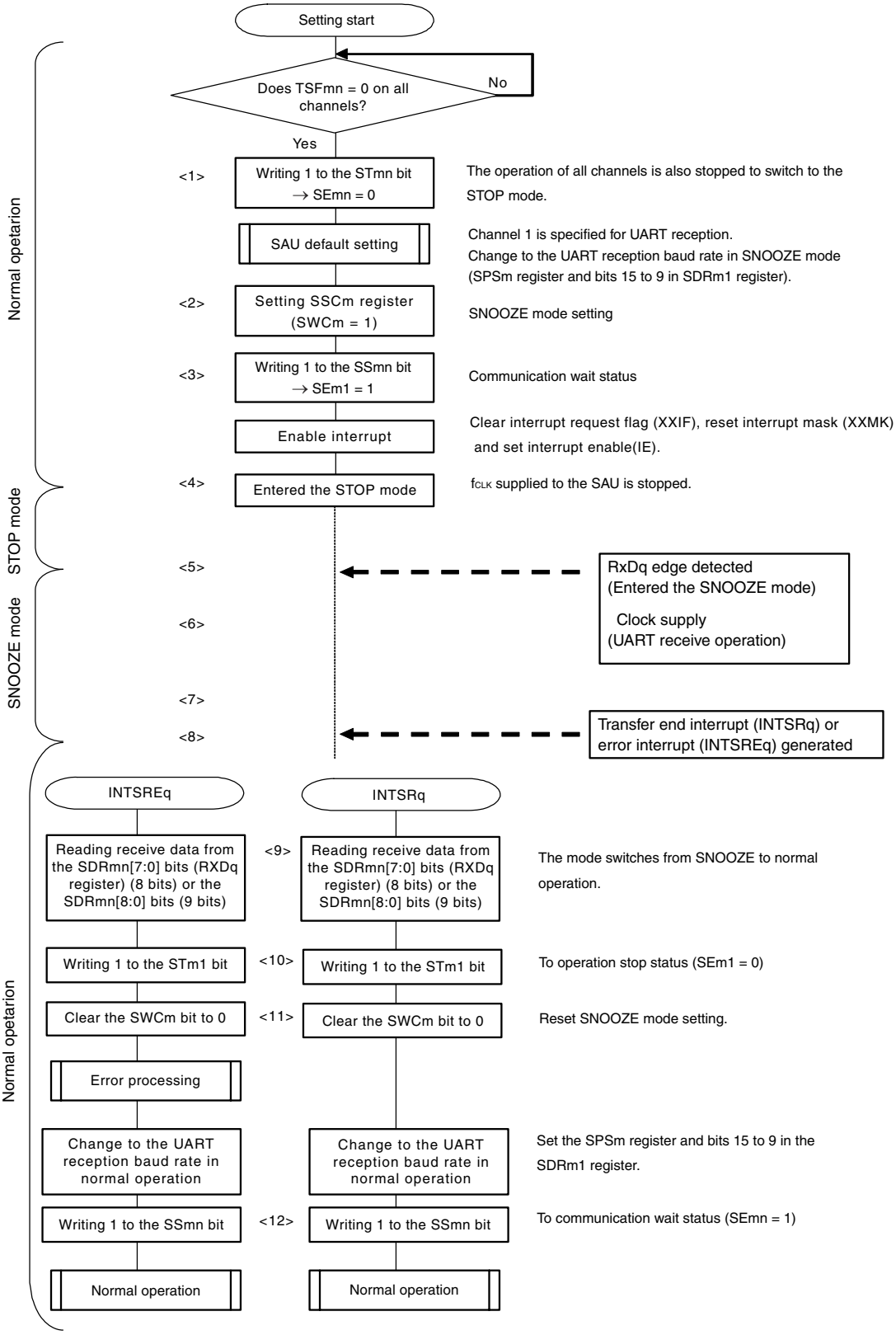



**Caution Before switching to the SNOOZE mode or after reception operation in the SNOOZE mode finishes, set the STm1 bit to 1 (clear the SEm1 bit, and stop the operation).**
**And after completion the receive operation, also clearing SWCm bit to 0 (SNOOZE mode release).**

**Remarks 1.** <1> to <11> in the figure correspond to <1> to <11> in **Figure 12-91 Flowchart of SNOOZE Mode Operation (EOCm1 = 0, SSECm = 0/1 or EOCm1 = 1, SSECm = 0)**.

**2.** m = 0

**Figure 12-91. Flowchart of SNOOZE Mode Operation (EOCm1 = 0, SSECm = 0/1 or EOCm1 = 1, SSECm = 0)**




<R>

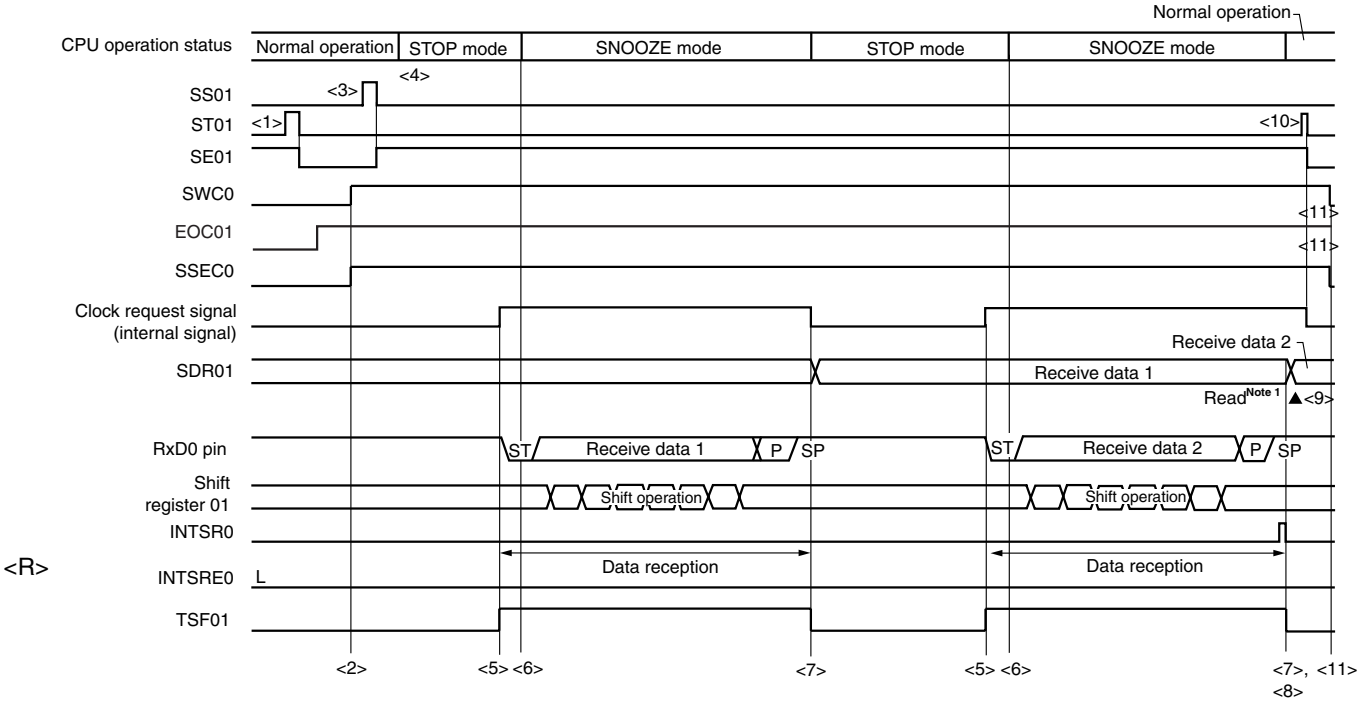
**Remarks 1.** <1> to <11> in the figure correspond to <1> to <11> in **Figure 12-89 Timing Chart of SNOOZE Mode Operation (EOCm1 = 0, SSECm = 0/1)** and **Figure 12-90 Timing Chart of SNOOZE Mode Operation (EOCm1 = 1, SSECm = 0)**.

**2.** m = 0; q = 0

### (3) SNOOZE mode operation (EOCm1 = 1, SSECm = 1: error interrupt (INTSRE0) generation stop)

Because EOCmn = 1, SSECm = 1, an error interrupt (INTSRE0) is not generated when a communication error occurs.

**Figure 12-92. Timing Chart of SNOOZE Mode Operation (Abnormal Operation <2>)**




**Note** Read the received data when SWCm = 1.

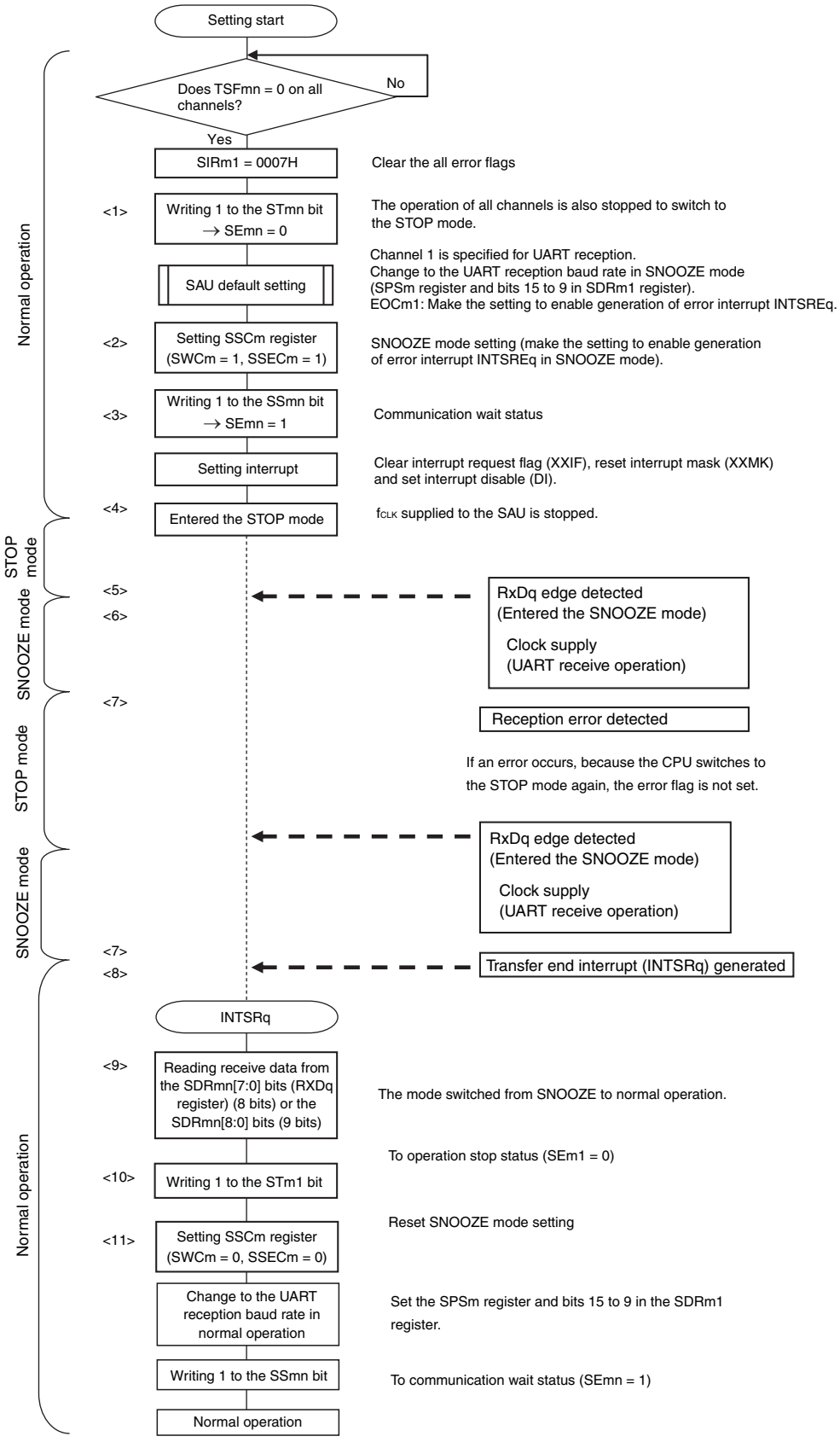
**Cautions 1. Before switching to the SNOOZE mode or after reception operation in the SNOOZE mode finishes, set the STm1 bit to 1 (clear the SEm1 bit, and stop the operation).**
**And after completion the receive operation, also clearing SWCm bit to 0 (SNOOZE mode release).**

**2. If the SSECm bit is 1, the PEFm1, FEFm1, and OVFm1 flags are not set when a parity error, framing error, or overrun error occurs and no error interrupt (INTSREq) is generated. Therefore, when the setting of SSECm = 1 is made, clear the PEFm1, FEFm1, or OVFm1 flag before setting the SWCm bit to 1 and read the value in SDRm1[7:0] (RxDq register) (8 bits) or SDRm1[8:0] (9 bits).**

**Remarks 1.** <1> to <9> in the figure correspond to <1> to <9> in **Figure 12-93 Flowchart of SNOOZE Mode Operation (EOCm1 = 1, SSECm = 1)**.

**2.** m = 0; q = 0

**Figure 12-93. Flowchart of SNOOZE Mode Operation (EOCm1 = 1, SSECm = 1)**




(**Caution** and **Remarks** are listed on the next page.)

**Caution If the SSECm bit is 1, the PEFm1, FEFm1, and OVFm1 flags are not set when a parity error, framing error, or overrun error occurs and no error interrupt (INTSREq) is generated. Therefore, when the setting of SSECm = 1 is made, clear the PEFm1, FEFm1, or OVFm1 flag before setting** <R> **the SWCm bit to 1 and read the value in SDRm1[7:0] (RxDq register) (8 bits) or SDRm1[8:0] (9 bits).**

**Remarks 1.** <1> to <9> in the figure correspond to <1> to <9> in **Figure 12-92 Timing Chart of SNOOZE Mode Operation (EOCm1 = 1, SSECm = 1)**.

**2.** m = 0; q = 0, n = 0 to 3

### 12.6.4 Calculating baud rate

#### (1) Baud rate calculation expression

The baud rate for UART (UART0 to UART2) communication can be calculated by the following expressions.

| (Baud rate) = {Operation clock ($f_{MCK}$) frequency of target channel} ÷ (SDRmn[15:9] + 1) ÷ 2 [bps] |
|---|

**Caution Setting serial data register mn (SDRmn) SDRmn[15:9] = (0000000B, 0000001B) is prohibited.**

**Remarks 1.** When UART is used, the value of SDRmn[15:9] is the value of bits 15 to 9 of the SDRmn register (0000010B to 1111111B) and therefore is 2 to 127.

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

The operation clock ($f_{MCK}$) is determined by serial clock select register m (SPSm) and bit 15 (CKSmn) of serial mode register mn (SMRmn).

**Table 12-4. Selection of Operation Clock For UART**

| SMRmn Register | SPSm Register | | | | | | | | Operation Clock ($f_{MCK}$)[Note] | |
|---|---|---|---|---|---|---|---|---|---|---|
| CKSmn | PRS m13 | PRS m12 | PRS m11 | PRS m10 | PRS m03 | PRS m02 | PRS m01 | PRS m00 | | $f_{CLK}$ = 32 MHz |
| 0 | X | X | X | X | 0 | 0 | 0 | 0 | $f_{CLK}$ | 32 MHz |
| | X | X | X | X | 0 | 0 | 0 | 1 | $f_{CLK}/2$ | 16 MHz |
| | X | X | X | X | 0 | 0 | 1 | 0 | $f_{CLK}/2^2$ | 8 MHz |
| | X | X | X | X | 0 | 0 | 1 | 1 | $f_{CLK}/2^3$ | 4 MHz |
| | X | X | X | X | 0 | 1 | 0 | 0 | $f_{CLK}/2^4$ | 2 MHz |
| | X | X | X | X | 0 | 1 | 0 | 1 | $f_{CLK}/2^5$ | 1 MHz |
| | X | X | X | X | 0 | 1 | 1 | 0 | $f_{CLK}/2^6$ | 500 kHz |
| | X | X | X | X | 0 | 1 | 1 | 1 | $f_{CLK}/2^7$ | 250 kHz |
| | X | X | X | X | 1 | 0 | 0 | 0 | $f_{CLK}/2^8$ | 125 kHz |
| | X | X | X | X | 1 | 0 | 0 | 1 | $f_{CLK}/2^9$ | 62.5 kHz |
| | X | X | X | X | 1 | 0 | 1 | 0 | $f_{CLK}/2^{10}$ | 31.25 kHz |
| | X | X | X | X | 1 | 0 | 1 | 1 | $f_{CLK}/2^{11}$ | 15.63 kHz |
| | X | X | X | X | 1 | 1 | 0 | 0 | $f_{CLK}/2^{12}$ | 7.81 kHz |
| | X | X | X | X | 1 | 1 | 0 | 1 | $f_{CLK}/2^{13}$ | 3.91 kHz |
| | X | X | X | X | 1 | 1 | 1 | 0 | $f_{CLK}/2^{14}$ | 1.95 kHz |
| | X | X | X | X | 1 | 1 | 1 | 1 | $f_{CLK}/2^{15}$ | 977 Hz |
| 1 | 0 | 0 | 0 | 0 | X | X | X | X | $f_{CLK}$ | 32 MHz |
| | 0 | 0 | 0 | 1 | X | X | X | X | $f_{CLK}/2$ | 16 MHz |
| | 0 | 0 | 1 | 0 | X | X | X | X | $f_{CLK}/2^2$ | 8 MHz |
| | 0 | 0 | 1 | 1 | X | X | X | X | $f_{CLK}/2^3$ | 4 MHz |
| | 0 | 1 | 0 | 0 | X | X | X | X | $f_{CLK}/2^4$ | 2 MHz |
| | 0 | 1 | 0 | 1 | X | X | X | X | $f_{CLK}/2^5$ | 1 MHz |
| | 0 | 1 | 1 | 0 | X | X | X | X | $f_{CLK}/2^6$ | 500 kHz |
| | 0 | 1 | 1 | 1 | X | X | X | X | $f_{CLK}/2^7$ | 250 kHz |
| | 1 | 0 | 0 | 0 | X | X | X | X | $f_{CLK}/2^8$ | 125 kHz |
| | 1 | 0 | 0 | 1 | X | X | X | X | $f_{CLK}/2^9$ | 62.5 kHz |
| | 1 | 0 | 1 | 0 | X | X | X | X | $f_{CLK}/2^{10}$ | 31.25 kHz |
| | 1 | 0 | 1 | 1 | X | X | X | X | $f_{CLK}/2^{11}$ | 15.63 kHz |
| | 1 | 1 | 0 | 0 | X | X | X | X | $f_{CLK}/2^{12}$ | 7.81 kHz |
| | 1 | 1 | 0 | 1 | X | X | X | X | $f_{CLK}/2^{13}$ | 3.91 kHz |
| | 1 | 1 | 1 | 0 | X | X | X | X | $f_{CLK}/2^{14}$ | 1.95 kHz |
| | 1 | 1 | 1 | 1 | X | X | X | X | $f_{CLK}/2^{15}$ | 977 Hz |
| Other than above | | | | | | | | | Setting prohibited | |

**Note** When changing the clock selected for $f_{CLK}$ (by changing the system clock control register (CKC) value), do so after having stopped (serial channel stop register m (STm) = 000FH) the operation of the serial array unit (SAU).

**Remarks 1.** X: Don't care

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

**(2) Baud rate error during transmission**

The baud rate error of UART (UART0 to UART2) communication during transmission can be calculated by the following expression. Make sure that the baud rate at the transmission side is within the permissible baud rate range at the reception side.

(Baud rate error) = (Calculated baud rate value) ÷ (Target baud rate) × 100 − 100 [%]

Here is an example of setting a UART baud rate at $f_{CLK}$ = 32 MHz.

| UART Baud Rate (Target Baud Rate) | $f_{CLK}$ = 32 MHz | | | |
|---|---|---|---|---|
| | Operation Clock ($f_{MCK}$) | SDRmn[15:9] | Calculated Baud Rate | Error from Target Baud Rate |
| 300 bps | $f_{CLK}/2^9$ | 103 | 300.48 bps | +0.16 % |
| 600 bps | $f_{CLK}/2^8$ | 103 | 600.96 bps | +0.16 % |
| 1200 bps | $f_{CLK}/2^7$ | 103 | 1201.92 bps | +0.16 % |
| 2400 bps | $f_{CLK}/2^6$ | 103 | 2403.85 bps | +0.16 % |
| 4800 bps | $f_{CLK}/2^5$ | 103 | 4807.69 bps | +0.16 % |
| 9600 bps | $f_{CLK}/2^4$ | 103 | 9615.38 bps | +0.16 % |
| 19200 bps | $f_{CLK}/2^3$ | 103 | 19230.8 bps | +0.16 % |
| 31250 bps | $f_{CLK}/2^3$ | 63 | 31250.0 bps | ±0.0 % |
| 38400 bps | $f_{CLK}/2^2$ | 103 | 38461.5 bps | +0.16 % |
| 76800 bps | $f_{CLK}/2$ | 103 | 76923.1 bps | +0.16 % |
| 153600 bps | $f_{CLK}$ | 103 | 153846 bps | +0.16 % |
| 312500 bps | $f_{CLK}$ | 50 | 313725.5 bps | +0.39 % |

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0, 2), mn = 00, 02, 10

**(3) Permissible baud rate range for reception**

The permissible baud rate range for reception during UART (UART0 to UART2) communication can be calculated by the following expression. Make sure that the baud rate at the transmission side is within the permissible baud rate range at the reception side.

$$\text{(Maximum receivable baud rate)} = \frac{2 \times k \times Nfr}{2 \times k \times Nfr - k + 2} \times Brate$$

$$\text{(Minimum receivable baud rate)} = \frac{2 \times k \times (Nfr - 1)}{2 \times k \times Nfr - k - 2} \times Brate$$

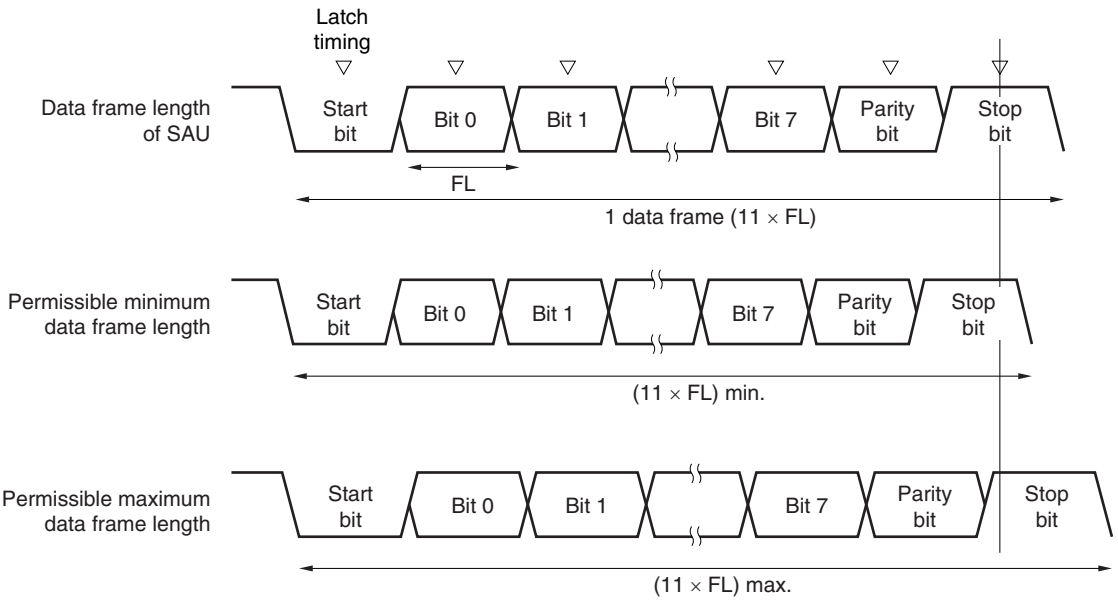
Brate: Calculated baud rate value at the reception side (See **12.6.4 (1) Baud rate calculation expression**.)

k: SDRmn[15:9] + 1

Nfr: 1 data frame length [bits]
= (Start bit) + (Data length) + (Parity bit) + (Stop bit)

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 1, 3), mn = 01, 03, 11

**Figure 12-94. Permissible Baud Rate Range for Reception (1 Data Frame Length = 11 Bits)**




As shown in Figure 12-94, the timing of latching receive data is determined by the division ratio set by bits 15 to 9 of serial data register mn (SDRmn) after the start bit is detected. If the last data (stop bit) is received before this latch timing, the data can be correctly received.

### 12.6.5 Procedure for processing errors that occurred during UART (UART0 to UART2) communication

The procedure for processing errors that occurred during UART (UART0 to UART2) communication is described in Figures 12-95 and 12-96.

**Figure 12-95. Processing Procedure in Case of Parity Error or Overrun Error**

| Software Manipulation | Hardware Status | Remark |
|---|---|---|
| Reads serial data register mn (SDRmn). → | The BFFmn bit of the SSRmn register is set to 0 and channel n is enabled to receive data. | This is to prevent an overrun error if the next reception is completed during error processing. |
| Reads serial status register mn (SSRmn). | | Error type is identified and the read value is used to clear error flag. |
| Writes 1 to serial flag clear trigger register mn (SIRmn). → | Error flag is cleared. | Error can be cleared only during reading, by writing the value read from the SSRmn register to the SIRmn register without modification. |

**Figure 12-96. Processing Procedure in Case of Framing Error**

| Software Manipulation | Hardware Status | Remark |
|---|---|---|
| Reads serial data register mn (SDRmn). → | The BFFmn bit of the SSRmn register is set to 0 and channel n is enabled to receive data. | This is to prevent an overrun error if the next reception is completed during error processing. |
| Reads serial status register mn (SSRmn). | | Error type is identified and the read value is used to clear error flag. |
| Writes serial flag clear trigger register mn (SIRmn). → | Error flag is cleared. | Error can be cleared only during reading, by writing the value read from the SSRmn register to the SIRmn register without modification. |
| Sets the STmn bit of serial channel stop register m (STm) to 1. → | The SEmn bit of serial channel enable status register m (SEm) is set to 0 and channel n stops operating. | |
| Synchronization with other party of communication | | Synchronization with the other party of communication is re-established and communication is resumed because it is considered that a framing error has occurred because the start bit has been shifted. |
| Sets the SSmn bit of serial channel start register m (SSm) to 1. → | The SEmn bit of serial channel enable status register m (SEm) is set to 1 and channel n is enabled to operate. | |

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

## 12.7 LIN Communication Operation

### 12.7.1 LIN transmission

Of UART transmission, UART2 of the 32, 48, and 64-pin products support LIN communication.

For LIN transmission, channel 0 of unit 1 is used.

| UART | UART0 | UART1 | UART2 |
|---|---|---|---|
| Support of LIN communication | Not supported | Not supported | Supported |
| Target channel | – | – | Channel 0 of SAU1 |
| Pins used | – | – | TxD2 |
| Interrupt | – | – | INTST2 |
| | Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected. | | |
| Error detection flag | None | | |
| Transfer data length | 8 bits | | |
| Transfer rate[Note] | Max. $f_{MCK}$/6 [bps] (SDR10 [15:9] = 2 or more), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps] | | |
| Data phase | Non-reverse output (default: high level)<br>Reverse output (default: low level) | | |
| Parity bit | No parity bit | | |
| Stop bit | Appending 1 bit | | |
| Data direction | LSB first | | |

**Note** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C), CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to +105°C)**). In addition, LIN communication is usually 2.4/9.6/19.2 kbps is often used.

**Remark** $f_{MCK}$: Operation clock frequency of target channel

$f_{CLK}$: System clock frequency

LIN stands for Local Interconnect Network and is a low-speed (1 to 20 kbps) serial communication protocol designed to reduce the cost of an automobile network.

Communication of LIN is single-master communication and up to 15 slaves can be connected to one master.

The slaves are used to control switches, actuators, and sensors, which are connected to the master via LIN.

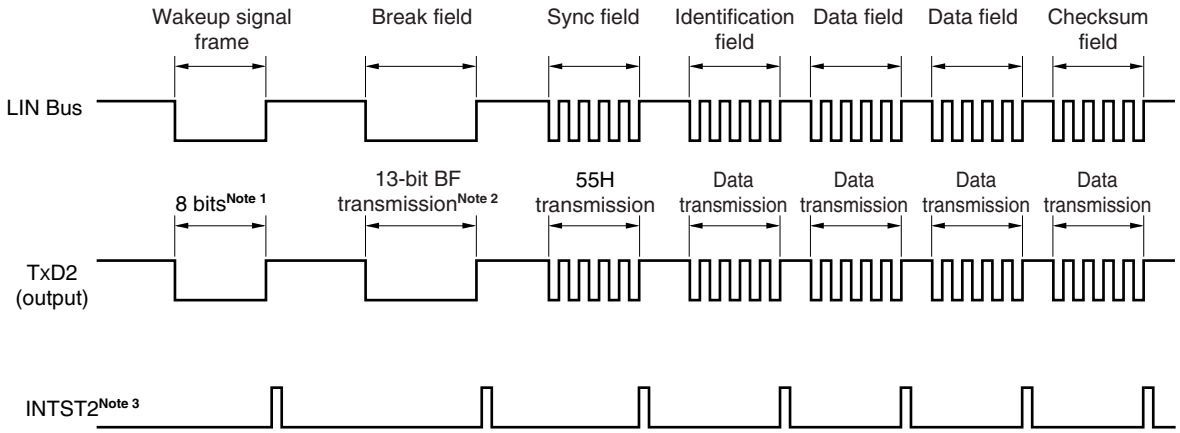
Usually, the master is connected to a network such as CAN (Controller Area Network).

A LIN bus is a single-wire bus to which nodes are connected via transceiver conforming to ISO9141.

According to the protocol of LIN, the master transmits a frame by attaching baud rate information to it. A slave receives this frame and corrects a baud rate error from the master. If the baud rate error of a slave is within ±15%, communication can be established.

Figure 12-97 outlines a master transmission operation of LIN.

**Figure 12-97. Master Transmission Operation of LIN**




<R> **Notes 1.** Set the baud rate in accordance with the wakeup signal regulations and transmit data of 80H.

**2.** A break field is defined to have a width of 13 bits and output a low level. Where the baud rate for main transfer is N [bps], therefore, the baud rate of the break field is calculated as follows.

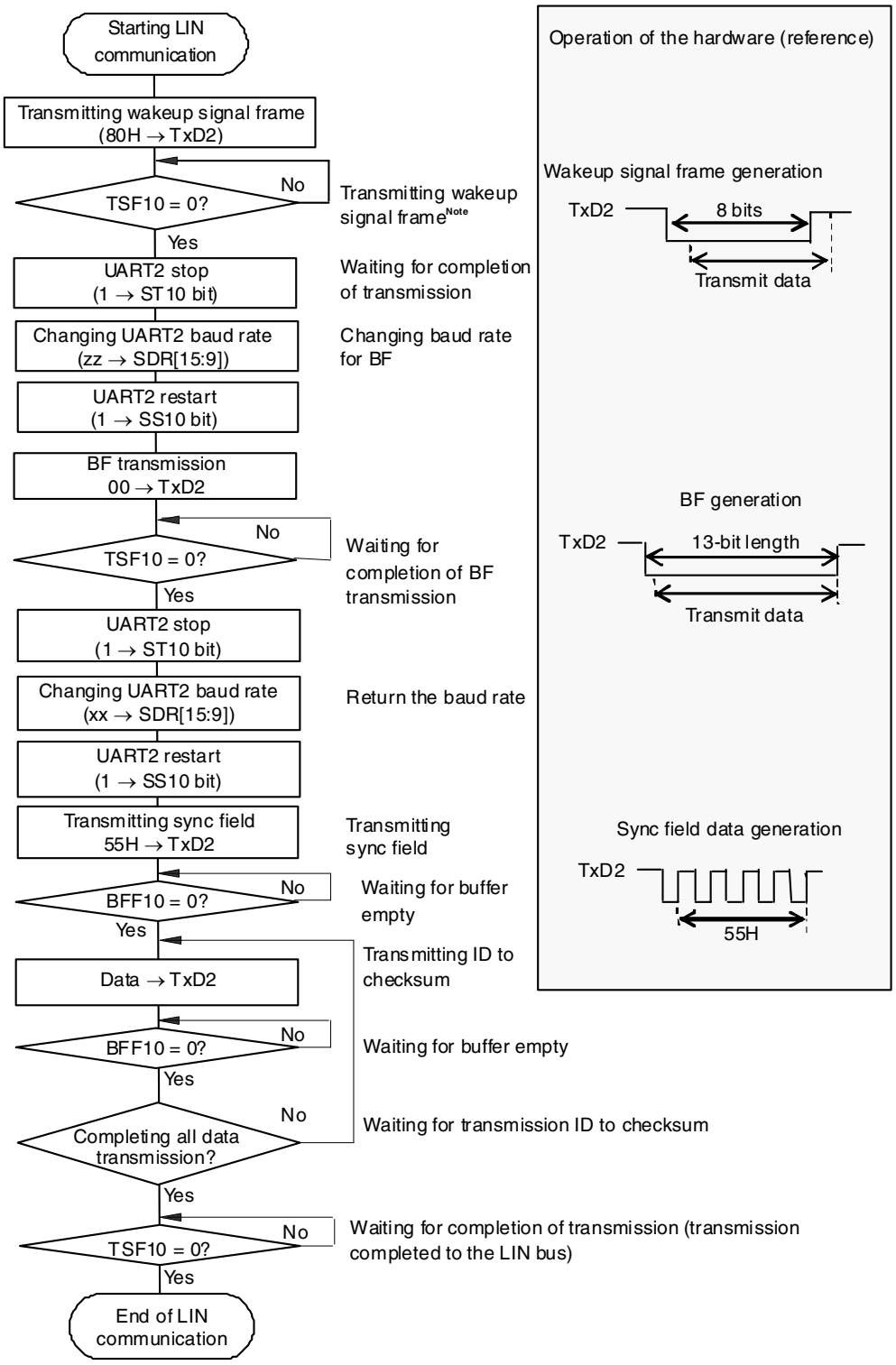
(Baud rate of break field) = 9/13 × N

By transmitting data of 00H at this baud rate, a break field is generated.

<R> **3.** INTST2 is output upon completion of transmission. INTST2 is also output at BF transmission.

**Remark** The interval between fields is controlled by software.

**Figure 12-98. Flowchart for LIN Transmission**




**Note** When LIN-bus start from sleep status only

**Remark** Default setting of the UART is complete, and the flow from the transmission enable status.

### 12.7.2 LIN reception

Of UART reception, UART2 of the 32, 48, and 64-pin products support LIN communication.

For LIN reception, channel 1 of unit 1 is used.

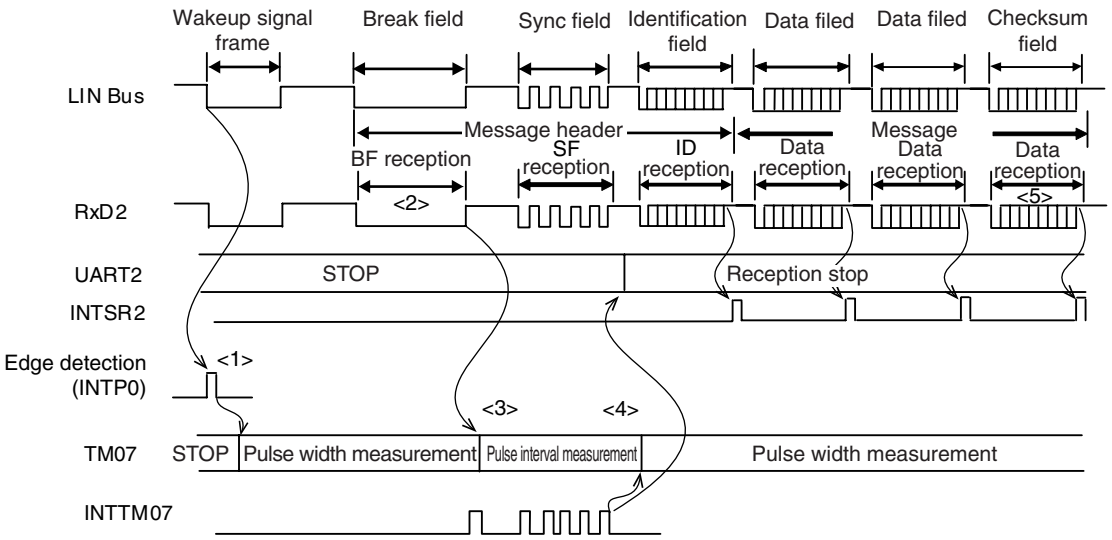
| UART | UART0 | UART1 | UART2 |
|---|---|---|---|
| Support of LIN communication | Not supported | Not supported | Supported |
| Target channel | – | – | Channel 1 of SAU1 |
| Pins used | – | – | RxD2 |
| Interrupt | – | – | INTSR2 |
| | Transfer end interrupt only (Setting the buffer empty interrupt is prohibited.) | | |
| Error interrupt | – | – | INTSRE2 |
| Error detection flag | • Framing error detection flag (FEF11)<br>• Parity error detection flag (PEF11)<br>• Overrun error detection flag (OVF11) | | |
| Transfer data length | 8 bits | | |
| Transfer rate[Note] | Max. $f_{MCK}$/6 [bps] (SDR11 [15:9] = 2 or more), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps] | | |
| Data phase | Non-reverse output (default: high level)<br>Reverse output (default: low level) | | |
| Parity bit | No parity bit (The parity bit is not checked.) | | |
| Stop bit | Check the first bit | | |
| Data direction | LSB first | | |

**Note** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C), CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to +105°C)**).

**Remark** $f_{MCK}$: Operation clock frequency of target channel
$f_{CLK}$: System clock frequency

Figure 12-99 outlines a reception operation of LIN.

**Figure 12-99. Reception Operation of LIN**




Here is the flow of reception processing.

<1> The wakeup signal is detected by detecting an interrupt edge (INTP0) on a pin. When the wakeup signal is detected, change TM07 to pulse width measurement upon detection of the wakeup signal to measure the low-level width of the BF signal. Then wait for BF signal reception.

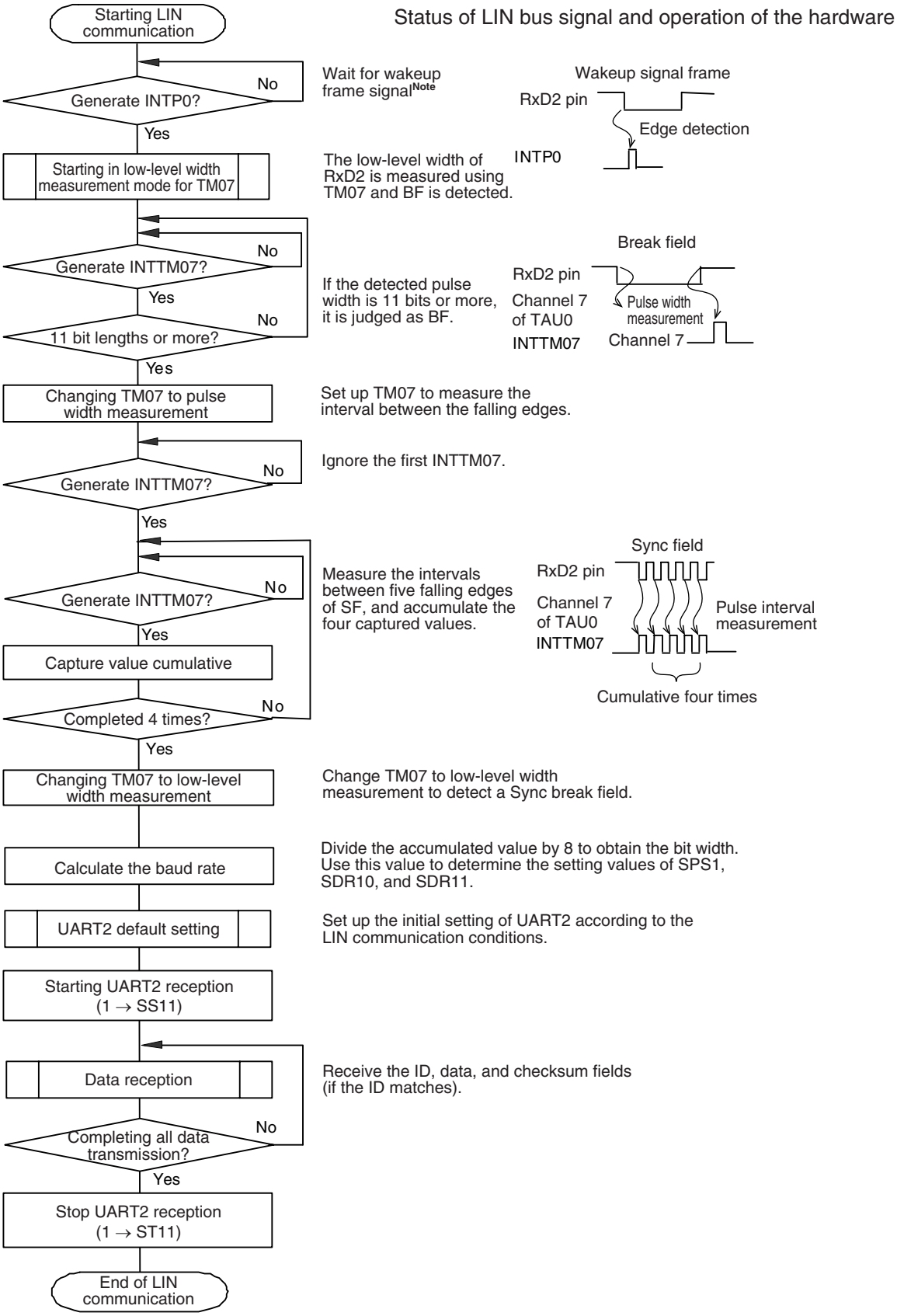
<2> TM07 starts measuring the low-level width upon detection of the falling edge of the BF signal, and then captures the data upon detection of the rising edge of the BF signal. The captured data is used to judge whether it is the BF signal.

<3> When the BF signal has been received normally, change TM07 to pulse interval measurement and measure the interval between the falling edges of the RxD2 signal in the sync field four times (see **6.8.4 Operation as input pulse interval measurement**).

<4> Calculate a baud rate error from the bit interval of sync field (SF). Stop UART2 once and adjust (re-set) the baud rate.

<5> The checksum field should be distinguished by software. In addition, processing to initialize UART2 after the checksum field is received and to wait for reception of BF should also be performed by software.

**Figure 12-100. Flowchart for LIN Reception**



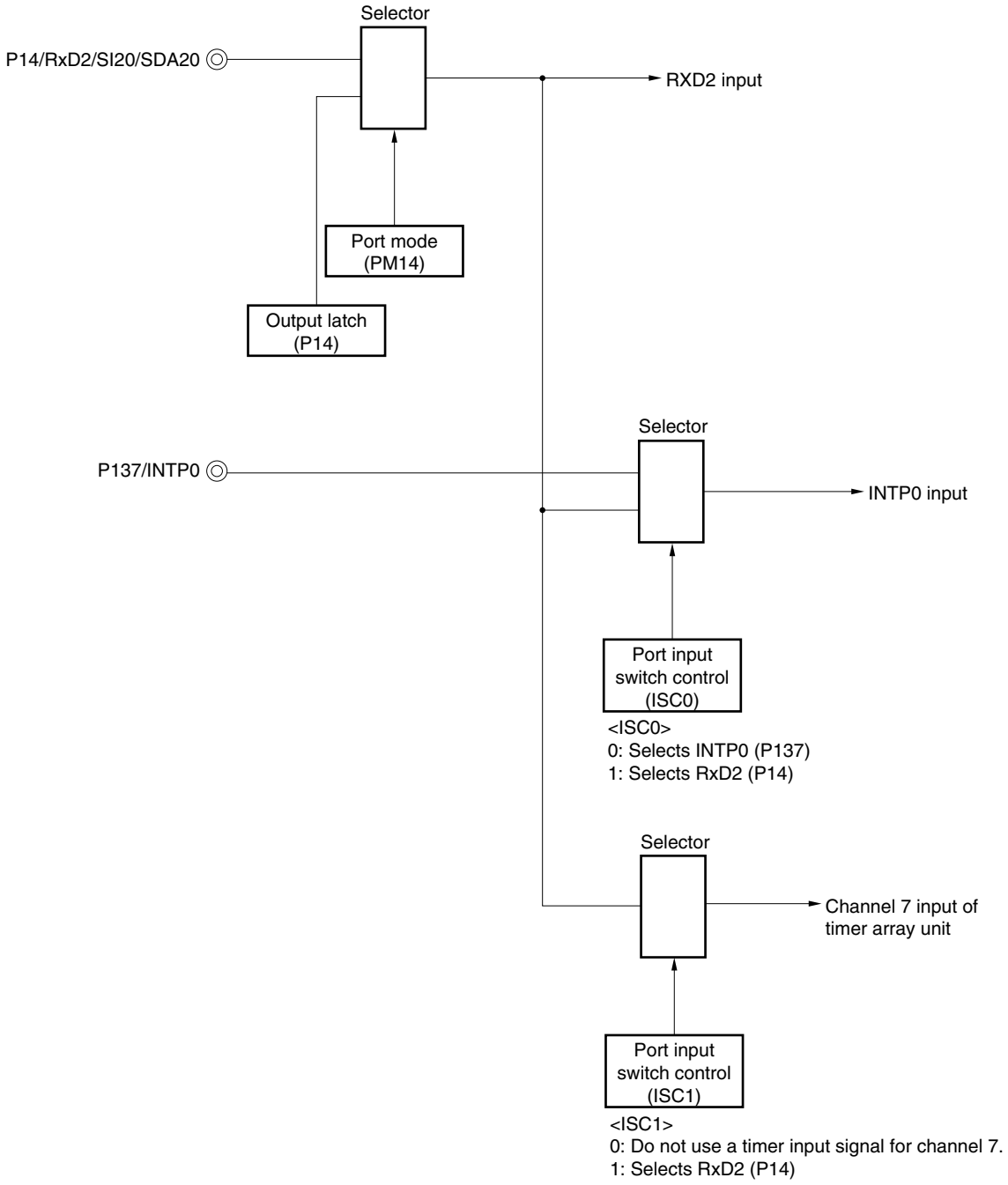

**Note** Required in the sleep status only.

Figure 12-101 and figure 12-102 show the configuration of a port that manipulates reception of LIN.

The wakeup signal transmitted from the master of LIN is received by detecting an edge of an external interrupt (INTP0). The length of the sync field transmitted from the master can be measured by using the external event capture operation of the timer array unit 0 to calculate a baud-rate error.

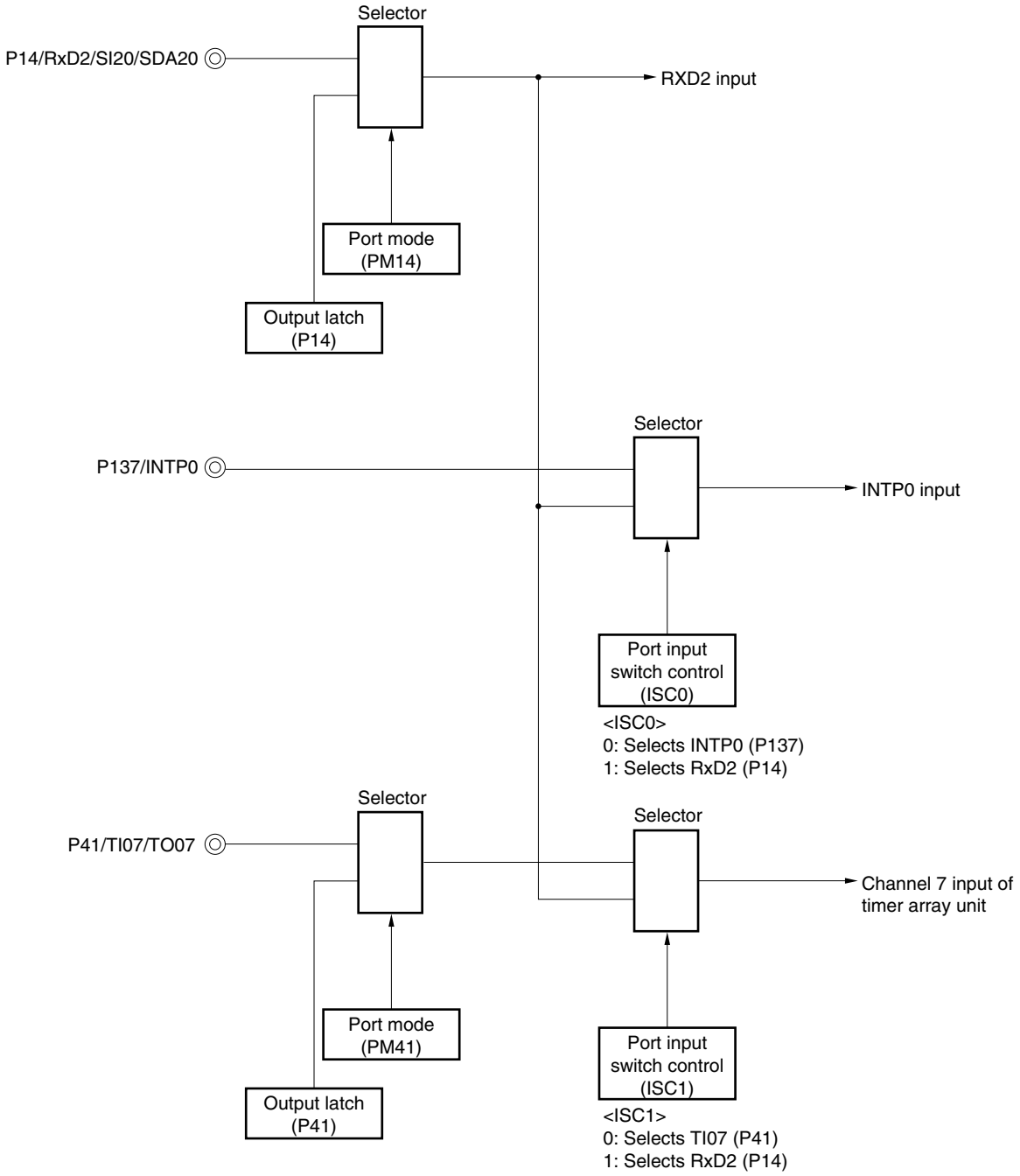
By controlling switch of port input (ISC0/ISC1), the input source of port input (RxD2) for reception can be input to the external interrupt pin (INTP0) and timer array unit

**Figure 12-101. Port Configuration for Manipulating Reception of LIN (32-pin)**




**Remark** ISC0, ISC1: Bits 0 and 1 of the input switch control register (ISC) (See **Figure 12-21**.)

**Figure 12-102. Port Configuration for Manipulating Reception of LIN (48, 64-pin)**




**Remark** ISC0, ISC1: Bits 0 and 1 of the input switch control register (ISC) (See **Figure 12-21**.)

The peripheral functions used for the LIN communication operation are as follows.

<Peripheral functions used>

- External interrupt (INTP0); Wakeup signal detection
  Usage: To detect an edge of the wakeup signal and the start of communication
- Channel 7 of timer array unit; Baud rate error detection, break field detection.
  Usage: To detect the length of the sync field (SF) and divide it by the number of bits in order to detect an error (The interval of the edge input to RxD2 is measured in the capture mode.)
  Measured the low-level width, determine whether break field (BF).
- Channels 0 and 1 (UART2) of serial array unit 1 (SAU1)

## 12.8 Operation of Simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) Communication

This is a clocked communication function to communicate with two or more devices by using two lines: serial clock (SCL) and serial data (SDA). This communication function is designed to execute single communication with devices such as EEPROM, flash memory, and A/D converter, and therefore, can be used only by the master.

Operate the control registers by software for setting the start and stop conditions while observing the specifications of the I²C bus line

[Data transmission/reception]

- Master transmission, master reception (only master function with a single master)
- ACK output function[Note] and ACK detection function
- Data length of 8 bits
  (When an address is transmitted, the address is specified by the higher 7 bits, and the least significant bit is used for R/W control.)
- Generation of start condition and stop condition for software

[Interrupt function]

- Transfer end interrupt

[Error detection flag]

- Overrun error
- ACK error

* [Functions not supported by simplified I²C]

- Slave transmission, slave reception
- Multi-master function (arbitration loss detection function)
- Wait detection function

**Note** When receiving the last data, ACK will not be output if 0 is written to the SOEmn (SOEm register) bit and serial communication data output is stopped. See the processing flow in **12.8.3 (2)** for details.

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

The channel supporting simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) is channels 0 to 3 of SAU0 and channel 0 and 1 of SAU1.

- 25-pin products

| Unit | Channel | Used as CSI | Used as UART | Used as Simplified I²C |
|---|---|---|---|---|
| 0 | 0 | CSI00 | UART0 | IIC00 |
| | 1 | – | | – |
| | 2 | – | UART1 | – |
| | 3 | CSI11 | | IIC11 |

- 32-pin products

<R>

| Unit | Channel | Used as CSI | Used as UART | Used as Simplified I²C |
|---|---|---|---|---|
| 0 | 0 | CSI00 | UART0 | IIC00 |
| | 1 | – | | – |
| | 2 | – | UART1 | – |
| | 3 | CSI11 | | IIC11 |
| 1 | 0 | CSI20 | UART2 (supporting LIN-bus) | IIC20 |
| | 1 | – | | – |

- 48-pin products

| Unit | Channel | Used as CSI | Used as UART | Used as Simplified I²C |
|---|---|---|---|---|
| 0 | 0 | CSI00 | UART0 | IIC00 |
| | 1 | CSI01 | | IIC01 |
| | 2 | – | UART1 | – |
| | 3 | CSI11 | | IIC11 |
| 1 | 0 | CSI20 | UART2 (supporting LIN-bus) | IIC20 |
| | 1 | CSI21 | | IIC21 |

- 64-pin products

| Unit | Channel | Used as CSI | Used as UART | Used as Simplified I²C |
|---|---|---|---|---|
| 0 | 0 | CSI00 | UART0 | IIC00 |
| | 1 | CSI01 | | IIC01 |
| | 2 | CSI10 | UART1 | IIC10 |
| | 3 | CSI11 | | IIC11 |
| 1 | 0 | CSI20 | UART2 (supporting LIN-bus) | IIC20 |
| | 1 | CSI21 | | IIC21 |

Simplified $I^2C$ (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) performs the following four types of communication operations.

- Address field transmission (See **12.8.1**.)
- Data transmission (See **12.8.2**.)
- Data reception (See **12.8.3**.)
- Stop condition generation (See **12.8.4**.)

### 12.8.1 Address field transmission

Address field transmission is a transmission operation that first executes in $I^2C$ communication to identify the target for transfer (slave). After a start condition is generated, an address (7 bits) and a transfer direction (1 bit) are transmitted in one frame.

| Simplified $I^2C$ | IIC00 | IIC01 | IIC10 | IIC11 | IIC20 | IIC21 |
|---|---|---|---|---|---|---|
| Target channel | Channel 0 of SAU0 | Channel 1 of SAU0 | Channel 2 of SAU0 | Channel 3 of SAU0 | Channel 0 of SAU1 | Channel 1 of SAU1 |
| Pins used | SCL00, SDA00Note 1 | SCL01, SDA01Note 1 | SCL10, SDA10Note 1 | SCL11, SDA11Note 1 | SCL20, SDA20Note 1 | SCL21, SDA21Note 1 |
| Interrupt | INTIIC00 | INTIIC01 | INTIIC10 | INTIIC11 | INTIIC20 | INTIIC21 |
| | Transfer end interrupt only (Setting the buffer empty interrupt is prohibited.) | | | | | |
| Error detection flag | ACK error detection flag (PEFmn) | | | | | |
| Transfer data length | 8 bits (transmitted with specifying the higher 7 bits as address and the least significant bit as R/W control) | | | | | |
| Transfer rateNote 2 | Max. $f_{MCK}$/4 [MHz] (SDRmn[15:9] = 1 or more) $f_{MCK}$: operation clock frequency of target channel<br>However, the following condition must be satisfied in each mode of $I^2C$.<br>• Max. 1 MHz (first mode plus)<br>• Max. 400 kHz (first mode)<br>• Max. 100 kHz (standard mode) | | | | | |
| Data level | Non-reversed output (default: high level) | | | | | |
| Parity bit | No parity bit | | | | | |
| Stop bit | Appending 1 bit (or ACK reception timing) | | | | | |
| Data direction | MSB first | | | | | |

<R> **Notes 1.** To perform communication via simplified $I^2C$, set the N-ch open-drain output ($V_{DD}$ tolerance (25- to 48-pin products/$EV_{DD}$ tolerance (64-pin products)) mode (POMxx = 1) for the port output mode registers (POMxx). See **4.3 Registers Controlling Port Function** and **4.5 Register Settings When Using Alternate Function** for details.

When IIC00, IIC10, IIC20 communicating with an external device with a different potential, set the N-ch open-drain output ($V_{DD}$ tolerance (25 to 48-pin products/$EV_{DD}$ tolerance (64-pin products)) mode (POMxx = 1) also for the clock input/output pins (SCL00, SCL10, SCL20). See **4.4.5 Handling different potential (1.8 V, 2.5 V) by using I/O buffers** for details.

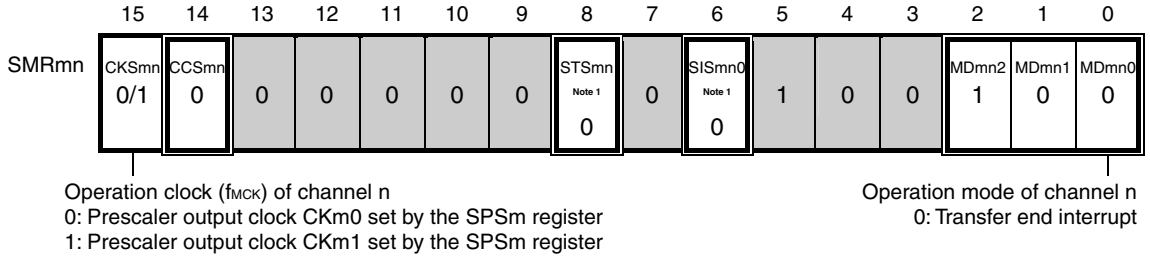
<R> **2.** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C)** and **CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to +105°C)**).

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

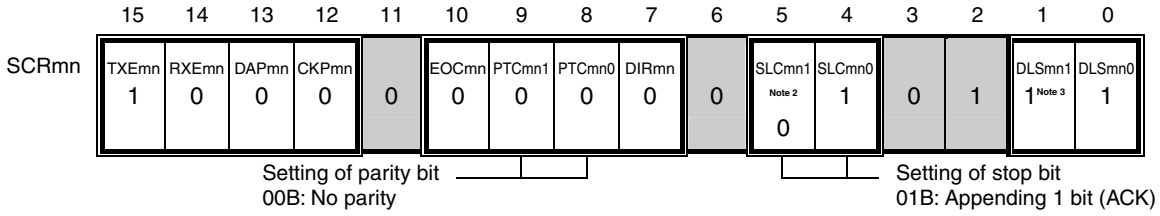
### (1) Register setting

**Figure 12-103. Example of Contents of Registers for Address Field Transmission of Simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) (1/2)**

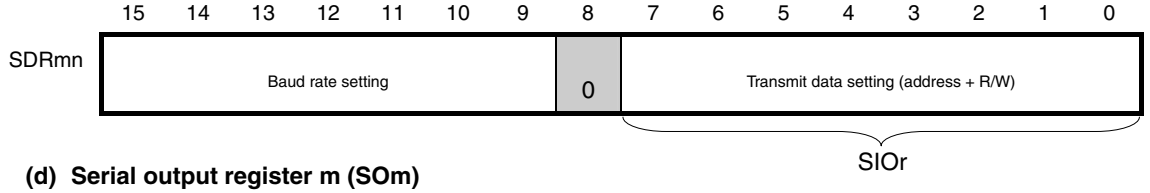
**(a) Serial mode register mn (SMRmn)**



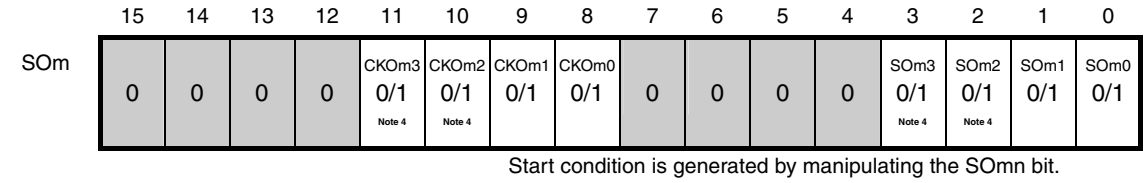

**(b) Serial communication operation setting register mn (SCRmn)**




**(c) Serial data register mn (SDRmn) (lower 8 bits: SIOr)**




**(d) Serial output register m (SOm)**




<R> **Notes 1.** Only provided for the SMR01, SMR03, and SMR11 registers.

<R> **2.** Only provided for the SCR00, SCR02, and SCR10 registers.

**3.** Only provided for the SCR00 and SCR01 registers. This bit is fixed to 1 for the other registers.

**4.** Unit 0 only

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), r: IIC number (r = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**2.** ☐: Setting is fixed in the IIC mode, ▒: Setting disabled (set to the initial value)
0/1: Set to 0 or 1 depending on the usage of the user

**Figure 12-103. Example of Contents of Registers for Address Field Transmission of Simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) (2/2)**

**(e) Serial output enable register m (SOEm)**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SOEm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SOEm3<br>0/1 Note | SOEm2<br>0/1 Note | SOEm1<br>0/1 | SOEm0<br>0/1 |

SOEmn = 0 until the start condition is generated, and SOEmn = 1 after generation.

**(f) Serial channel start register m (SSm) … Sets only the bits of the target channel is 1.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SSm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SSm3<br>0/1 Note | SSm2<br>0/1 Note | SSm1<br>0/1 | SSm0<br>0/1 |

SSmn = 0 until the start condition is generated, and SSmn = 1 after generation.

**Note** Unit 0 only

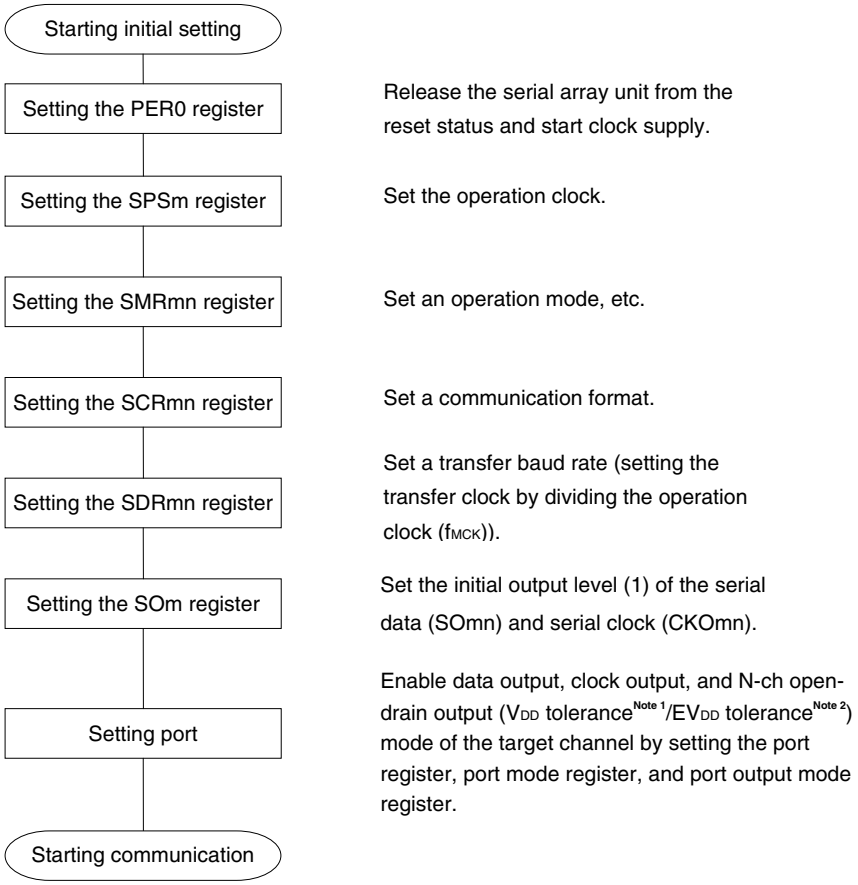
**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), r: IIC number (r = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**2.** ☐ : Setting disabled (set to the initial value)

0/1: Set to 0 or 1 depending on the usage of the user

### (2) Operation procedure

**Figure 12-104. Initial Setting Procedure for Address Field Transmission**



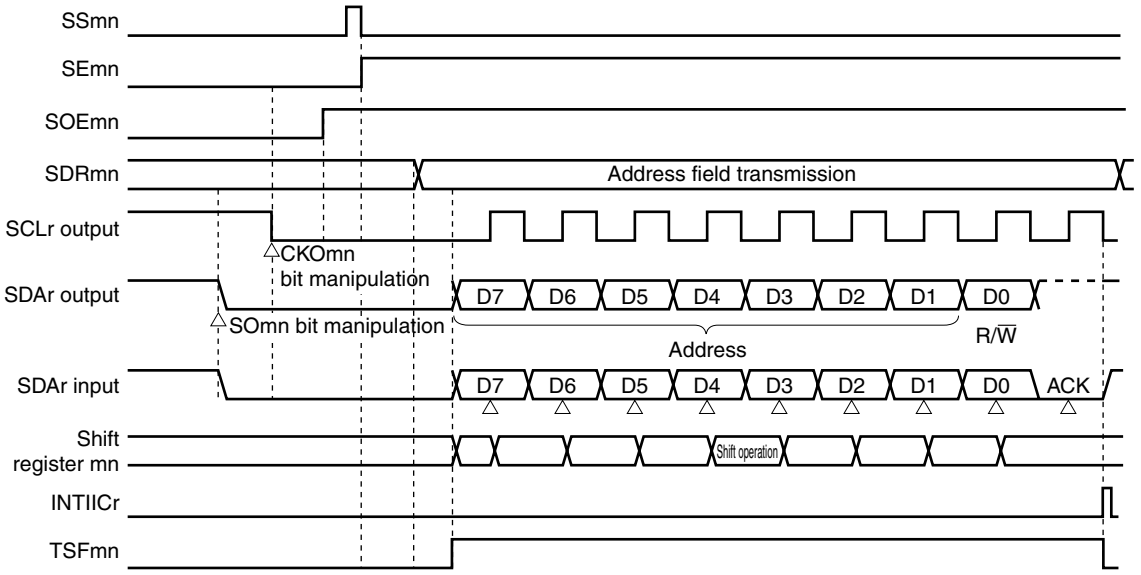

**Notes 1.** 25 to 48-pin products

**2.** 64-pin products

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

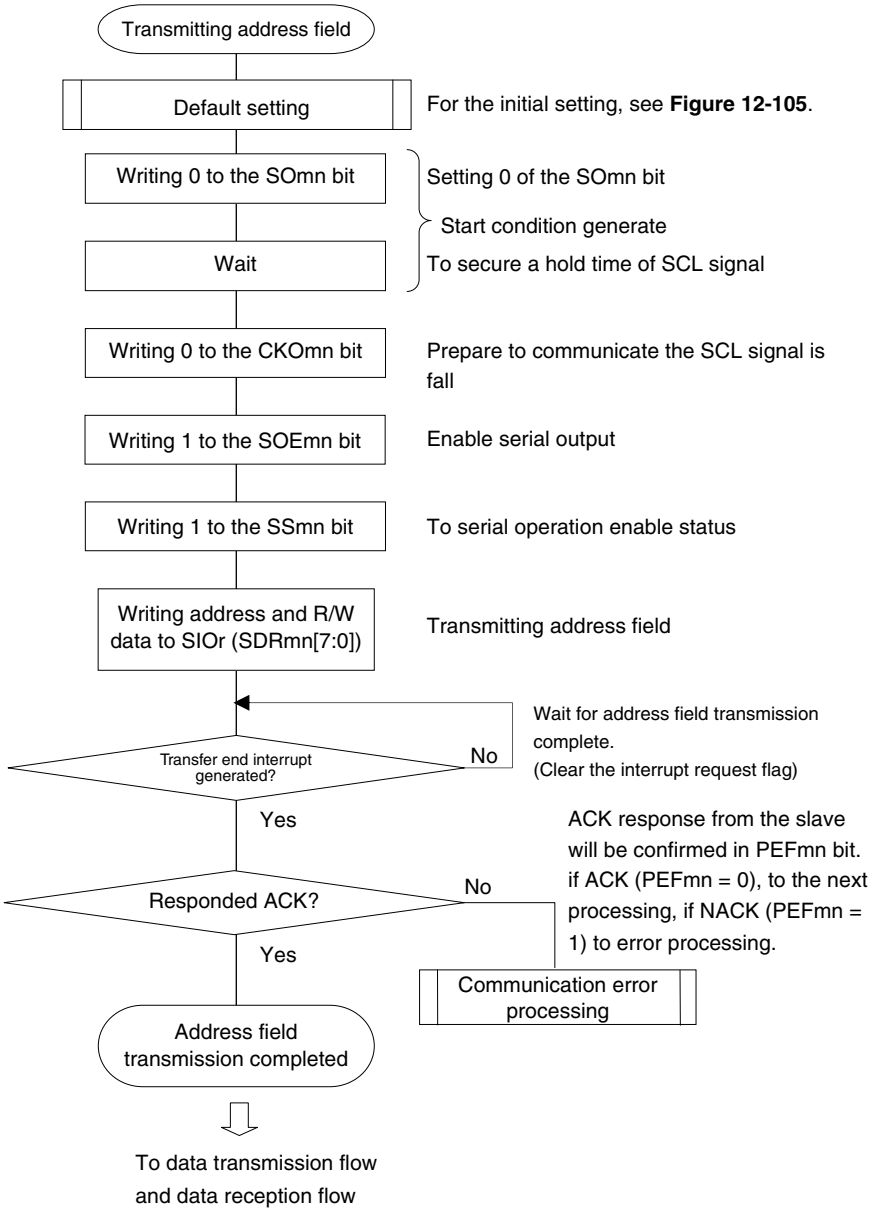
### (3) Processing flow

**Figure 12-105. Timing Chart of Address Field Transmission**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), r: IIC number (r = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**Figure 12-106. Flowchart of Simplified I²C Address Field Transmission**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), r: IIC number (r = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

### 12.8.2 Data transmission

Data transmission is an operation to transmit data to the target for transfer (slave) after transmission of an address field. After all data are transmitted to the slave, a stop condition is generated and the bus is released.

| Simplified $I^2C$ | IIC00 | IIC01 | IIC10 | IIC11 | IIC20 | IIC21 |
|---|---|---|---|---|---|---|
| Target channel | Channel 0 of SAU0 | Channel 1 of SAU0 | Channel 2 of SAU0 | Channel 3 of SAU0 | Channel 0 of SAU1 | Channel 1 of SAU1 |
| Pins used | SCL00, SDA00Note 1 | SCL01, SDA01Note 1 | SCL10, SDA10Note 1 | SCL11, SDA11Note 1 | SCL20, SDA20Note 1 | SCL21, SDA21Note 1 |
| Interrupt | INTIIC00 | INTIIC01 | INTIIC10 | INTIIC11 | INTIIC20 | INTIIC21 |
| | Transfer end interrupt only (Setting the buffer empty interrupt is prohibited.) | | | | | |
| Error detection flag | ACK error detection flag (PEFmn) | | | | | |
| Transfer data length | 8 bits | | | | | |
| Transfer rateNote 2 | Max. $f_{MCK}$/4 [MHz] (SDRmn[15:9] = 1 or more) $f_{MCK}$: operation clock frequency of target channel<br>However, the following condition must be satisfied in each mode of $I^2C$.<br>• Max. 1 MHz (first mode plus)<br>• Max. 400 kHz (first mode)<br>• Max. 100 kHz (standard mode) | | | | | |
| <R> Data level | Non-reversed output (default: high level) | | | | | |
| Parity bit | No parity bit | | | | | |
| Stop bit | Appending 1 bit (for ACK reception timing) | | | | | |
| Data direction | MSB first | | | | | |

<R> **Notes 1.** To perform communication via simplified $I^2C$, set the N-ch open-drain output ($V_{DD}$ tolerance (25- to 48-pin products/$EV_{DD}$ tolerance (64-pin products)) mode (POMxx = 1) for the port output mode registers (POMxx). See **4.3 Registers Controlling Port Function** and **4.5 Register Settings When Using Alternate Function** for details.

When IIC00, IIC10, IIC20 communicating with an external device with a different potential, set the N-ch open-drain output ($V_{DD}$ tolerance (25 to 48-pin products/$EV_{DD}$ tolerance (64-pin products)) mode (POMxx = 1) also for the clock input/output pins (SCL00, SCL10, SCL20).

See **4.4.5 Handling different potential (1.8 V, 2.5 V) by using I/O buffers** for details.

<R> **2.** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C)** and **CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to +105°C)**).

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

**(1) Register setting**

**Figure 12-107. Example of Contents of Registers for Data Transmission of Simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) (1/2)**

**(a) Serial mode register mn (SMRmn) … Do not manipulate this register during data transmission/reception.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SMRmn | CKSmn | CCSmn | 0 | 0 | 0 | 0 | 0 | STSmn | 0 | SISmn0 | 1 | 0 | 0 | MDmn2 | MDmn1 | MDmn0 |
| | 0/1 | 0 | | | | | | 0Note 1 | | 0Note 1 | | | | 1 | 0 | 0 |

**(b) Serial communication operation setting register mn (SCRmn) … Do not manipulate the bits of this register, except the TXEmn and RXEmn bits, during data transmission/reception.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SCRmn | TXEmn | RXEmn | DAPmn | CKPmn | 0 | EOCmn | PTCmn1 | PTCmn0 | DIRmn | 0 | SLCmn1 | SLCmn0 | 0 | 1 | DLSmn1 | DLSmn0 |
| | 1 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | | 0Note 2 | 1 | | | 1Note 3 | 1 |

**(c) Serial data register mn (SDRmn) (lower 8 bits: SIOr) … During data transmission/reception, valid only lower 8-bits (SIOr)**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SDRmn | Baud rate settingNote 4 | | | | | | | 0 | Transmit data setting | | | | | | | |
| | | | | | | | | | SIOr | | | | | | | |

**(d) Serial output register m (SOm) … Do not manipulate this register during data transmission/reception.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SOm | 0 | 0 | 0 | 0 | CKOm3 Note 5 0/1 Note 6 | CKOm2 Note 5 0/1 Note 6 | CKOm1 0/1 Note 6 | CKOm0 0/1 Note 6 | 0 | 0 | 0 | 0 | SOm3 Note 5 0/1 Note 6 | SOm2 Note 5 0/1 Note 6 | SOm1 0/1 Note 6 | SOm0 0/1 Note 6 |

<R> **Notes 1.** Only provided for the SMR01, SMR03, and SMR11 registers.

<R> **2.** Only provided for the SCR00, SCR02, and SCR10 registers.

**3.** Only provided for the SCR00 and SCR01 registers. This bit is fixed to 1 for the other registers.

**4.** Because the setting is completed by address field transmission, setting is not required.

**5.** Unit 0 only.

**6.** The value varies depending on the communication data during communication operation.

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), r: IIC number (r = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**2.** □: Setting is fixed in the IIC mode, ■: Setting disabled (set to the initial value)
0/1: Set to 0 or 1 depending on the usage of the user

**Figure 12-107. Example of Contents of Registers for Data Transmission of Simplified $I^2C$ (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) (2/2)**

**(e) Serial output enable register m (SOEm) … Do not manipulate this register during data transmission/reception.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SOEm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SOEm3 Note 1 | SOEm2 Note 1 | SOEm1 1 | SOEm0 1 |

**(f) Serial channel start register m (SSm) … Do not manipulate this register during data transmission/reception.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SSm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SSm3 Note 0/1 | SSm2 Note 0/1 | SSm1 0/1 | SSm0 0/1 |

**Note** Unit 0 only.

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

**2.** ☐: Setting is fixed in the IIC mode, ▒: Setting disabled (set to the initial value)
0/1: Set to 0 or 1 depending on the usage of the user

### (2) Processing flow

**Figure 12-108. Timing Chart of Data Transmission**

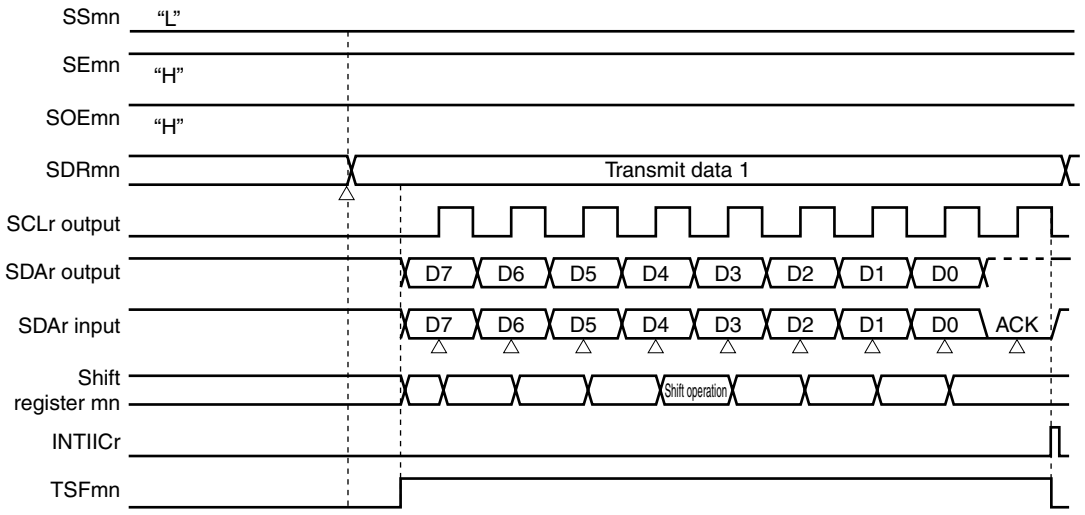
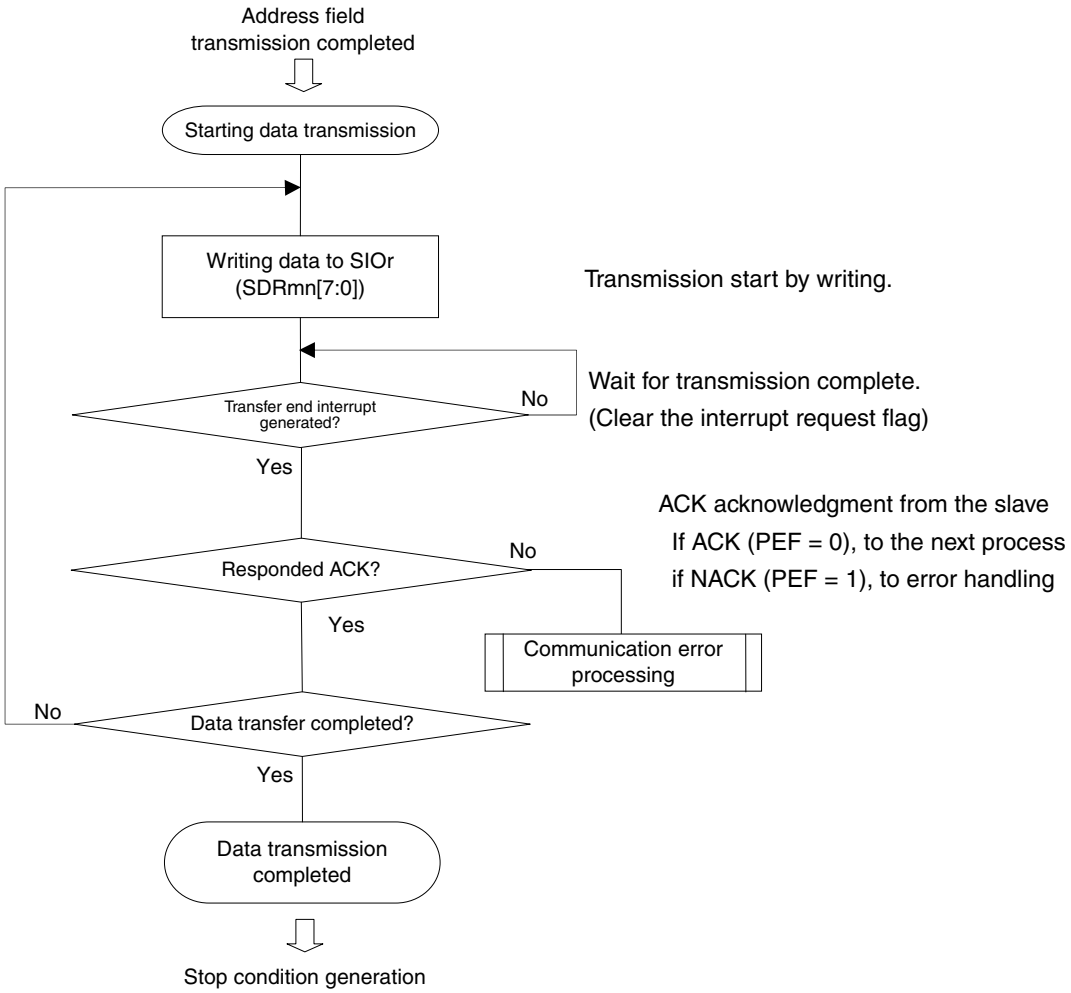



**Figure 12-109. Flowchart of Simplified I²C Data Transmission**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), r: IIC number (r = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

### 12.8.3 Data reception

Data reception is an operation to receive data to the target for transfer (slave) after transmission of an address field. After all data are received to the slave, a stop condition is generated and the bus is released.

| Simplified $I^2C$ | IIC00 | IIC01 | IIC10 | IIC11 | IIC20 | IIC21 |
|---|---|---|---|---|---|---|
| Target channel | Channel 0 of SAU0 | Channel 1 of SAU0 | Channel 2 of SAU0 | Channel 3 of SAU0 | Channel 0 of SAU1 | Channel 1 of SAU1 |
| Pins used | SCL00, SDA00Note 1 | SCL01, SDA01Note 1 | SCL10, SDA10Note 1 | SCL11, SDA11Note 1 | SCL20, SDA20Note 1 | SCL21, SDA21Note 1 |
| Interrupt | INTIIC00 | INTIIC01 | INTIIC10 | INTIIC11 | INTIIC20 | INTIIC21 |
| | Transfer end interrupt only (Setting the buffer empty interrupt is prohibited.) | | | | | |
| Error detection flag | Overrun error detection flag (OVFmn) only | | | | | |
| Transfer data length | 8 bits | | | | | |
| Transfer rateNote 2 | Max. $f_{MCK}$/4 [MHz] (SDRmn[15:9] = 1 or more) $f_{MCK}$: operation clock frequency of target channel<br>However, the following condition must be satisfied in each mode of $I^2C$.<br>• Max. 1 MHz (first mode plus)<br>• Max. 400 kHz (first mode)<br>• Max. 100 kHz (standard mode) | | | | | |
| Data level | Non-reversed output (default: high level) | | | | | |
| Parity bit | No parity bit | | | | | |
| Stop bit | Appending 1 bit (ACK transmission) | | | | | |
| Data direction | MSB first | | | | | |

<R> **Notes 1.** To perform communication via simplified $I^2C$, set the N-ch open-drain output ($V_{DD}$ tolerance (25 to 48-pin products/$EV_{DD}$ tolerance (64-pin products)) mode (POMxx = 1) for the port output mode registers (POMxx). See **4.3 Registers Controlling Port Function** and **4.5 Register Settings When Using Alternate Function** for details.

When IIC00, IIC10, IIC20 communicating with an external device with a different potential, set the N-ch open-drain output ($V_{DD}$ tolerance (25 to 48-pin products/$EV_{DD}$ tolerance (64-pin products)) mode (POMxx = 1) also for the clock input/output pins (SCL00, SCL10, SCL20).

See **4.4.5 Handling different potential (1.8 V, 2.5 V) by using I/O buffers** for details.

<R> **2.** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C)** and **CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to +105°C)**).

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

### (1) Register setting

**Figure 12-110. Example of Contents of Registers for Data Reception of Simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) (1/2)**

**(a) Serial mode register mn (SMRmn) … Do not manipulate this register during data transmission/reception.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SMRmn | CKSmn | CCSmn | | | | | | STSmn | | SISmn0 | | | | MDmn2 | MDmn1 | MDmn0 |
| | 0/1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 Note 1 | 0 | 0 Note 1 | 1 | 0 | 0 | 1 | 0 | 0 |

**(b) Serial communication operation setting register mn (SCRmn) … Do not manipulate the bits of this register, except the TXEmn and RXEmn bits, during data transmission/reception.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SCRmn | TXEmn | RXEmn | DAPmn | CKPmn | | EOCmn | PTCmn1 | PTCmn0 | DIRmn | | SLCmn1 | SLCmn0 | | | DLSmn1 | DLSmn0 |
| | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 Note 2 | 1 | 0 | 1 | 1 Note 3 | 1 |

**(c) Serial data register mn (SDRmn) (lower 8 bits: SIOr)**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SDRmn | Baud rate setting Note 4 | | | | | | | 0 | Dummy transmit data setting (FFH) | | | | | | | |

SIOr

**(d) Serial output register m (SOm) … Do not manipulate this register during data transmission/reception.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SOm | 0 | 0 | 0 | 0 | CKOm3 Note 5 | CKOm2 Note 5 | CKOm1 | CKOm0 | 0 | 0 | 0 | 0 | SOm3 Note 5 | SOm2 Note 5 | SOm1 | SOm0 |
| | | | | | 0/1 Note 6 | 0/1 Note 6 | 0/1 Note 6 | 0/1 Note 6 | | | | | 0/1 Note 6 | 0/1 Note 6 | 0/1 Note 6 | 0/1 Note 6 |

<R> **Notes 1.** Only provided for the SMR01, SMR03, and SMR11 registers.

<R> **2.** Only provided for the SCR00, SCR02, and SCR10 registers.

**3.** Only provided for the SCR00 and SCR01 registers. This bit is fixed to 1 for the other registers.

**4.** Because the setting is completed by address field transmission, setting is not required.

**5.** Unit 0 only.

**6.** The value varies depending on the communication data during communication operation.

**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), r: IIC number (r = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**2.** ☐: Setting is fixed in the IIC mode, ▒: Setting disabled (set to the initial value)
0/1: Set to 0 or 1 depending on the usage of the user

**Figure 12-110. Example of Contents of Registers for Data Reception of Simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) (2/2)**

**(e) Serial output enable register m (SOEm) … Do not manipulate this register during data transmission/reception.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SOEm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SOEm3 Note 0/1 | SOEm2 Note 0/1 | SOEm1 0/1 | SOEm0 0/1 |

**(f) Serial channel start register m (SSm) … Do not manipulate this register during data transmission/reception.**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SSm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SSm3 Note 0/1 | SSm2 Note 0/1 | SSm1 0/1 | SSm0 0/1 |

**Note** Unit 0 only

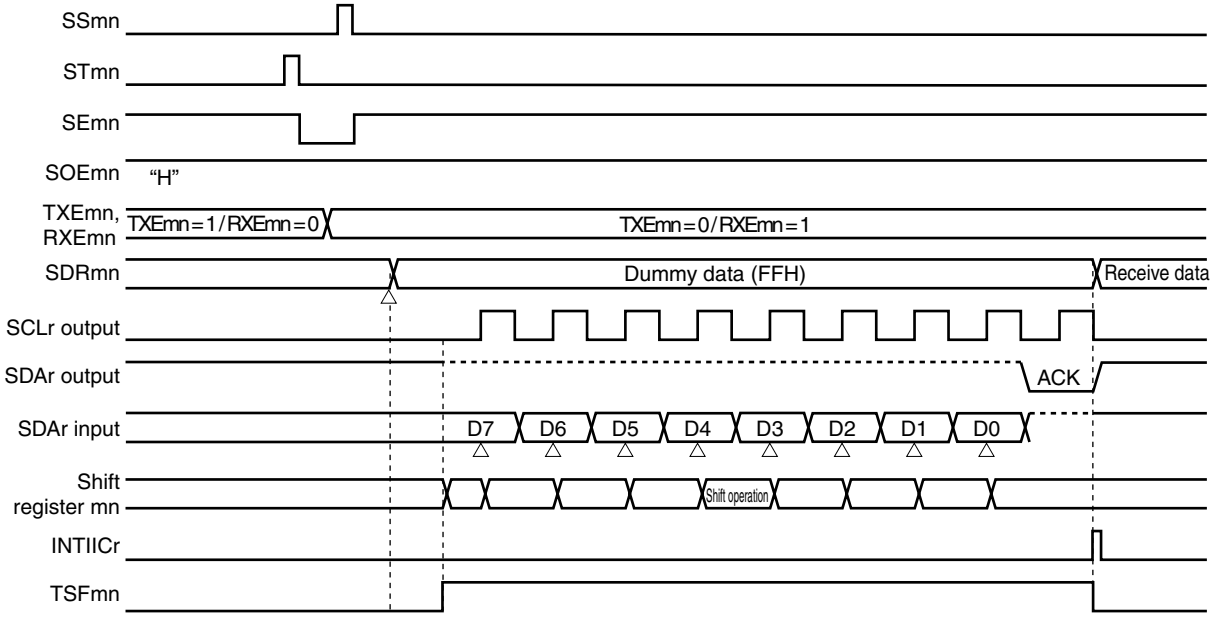
**Remarks 1.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

**2.** ☐: Setting is fixed in the CSI master transmission mode, ▒: Setting disabled (set to the initial value)
0/1: Set to 0 or 1 depending on the usage of the user

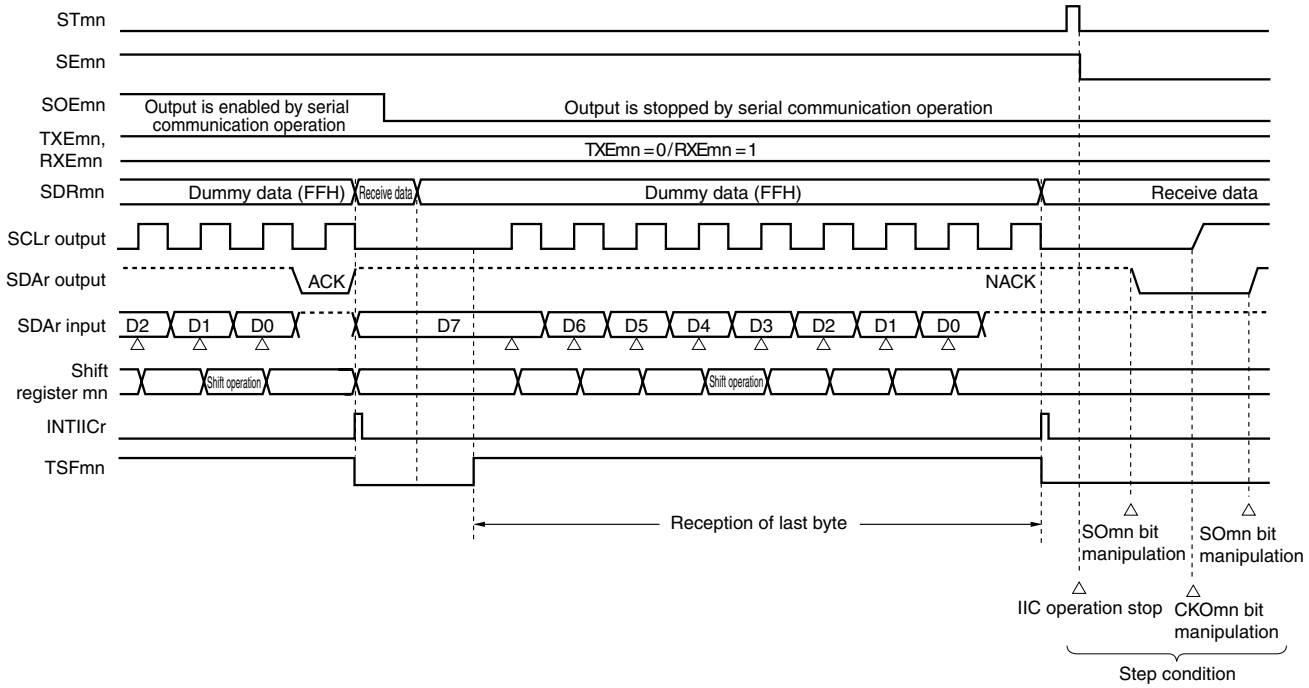
### (2) Processing flow

**Figure 12-111. Timing Chart of Data Reception**

**(a) When starting data reception**

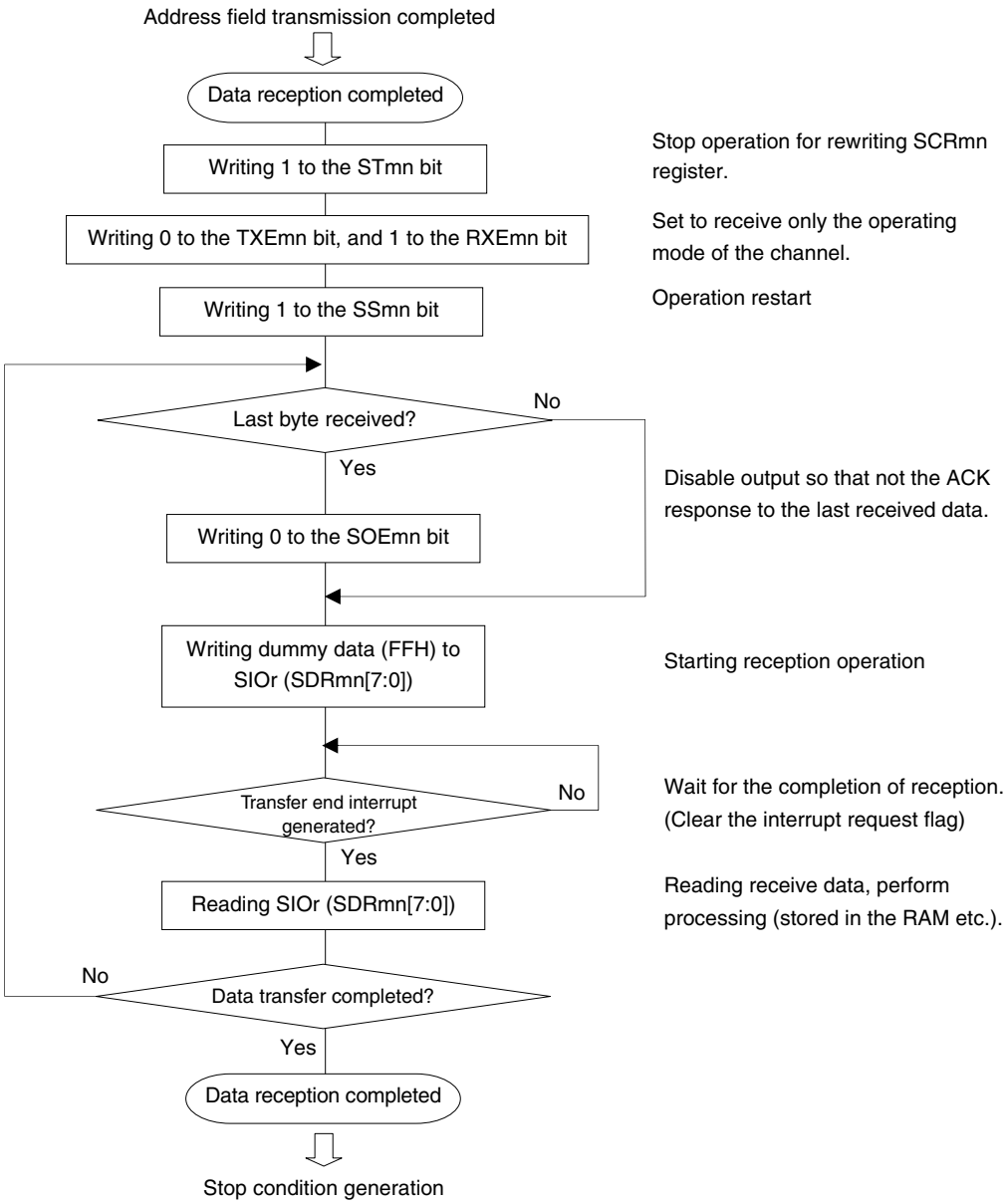



**(b) When receiving last data**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), r: IIC number (r = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

**Figure 12-112. Flowchart of Data Reception**




**Caution ACK is not output when the last data is received (NACK). Communication is then completed by setting "1" to the STmn bit of serial channel stop register m (STm) to stop operation and generating a stop condition.**

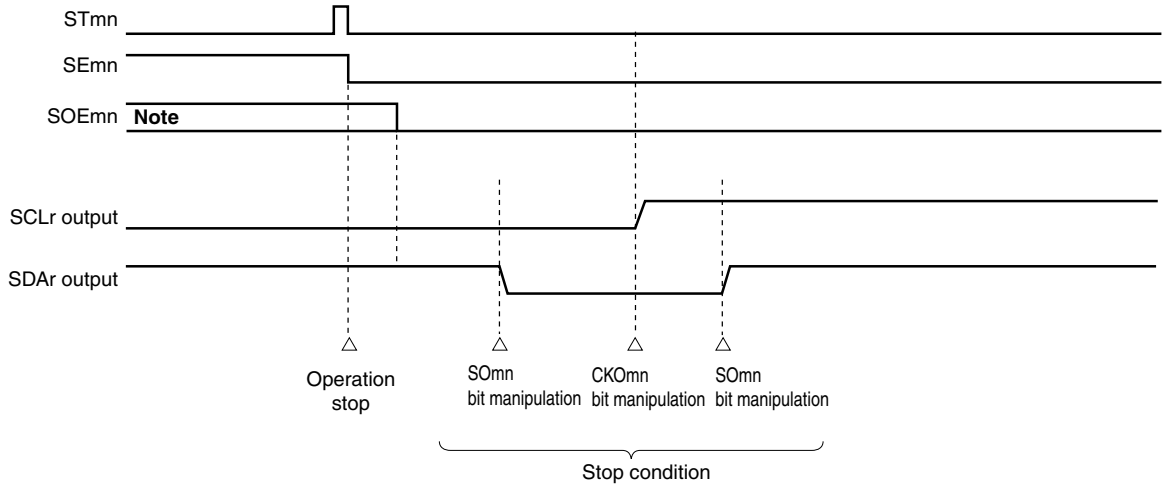
**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), r: IIC number (r = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

### 12.8.4 Stop condition generation

After all data are transmitted to or received from the target slave, a stop condition is generated and the bus is released.

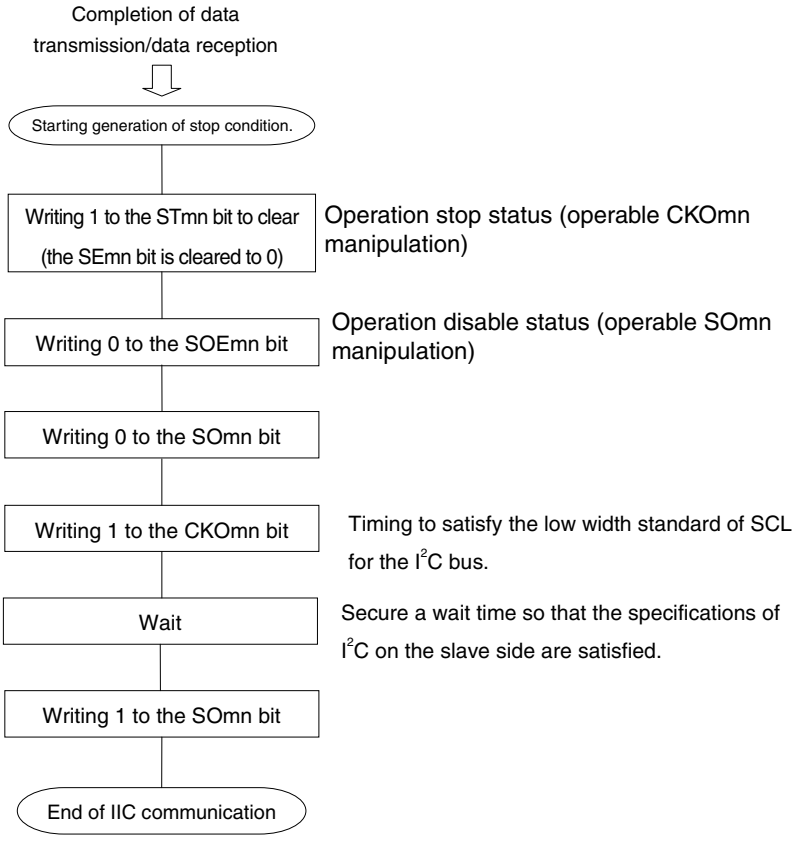
#### (1) Processing flow

**Figure 12-113. Timing Chart of Stop Condition Generation**




**Note** During a receive operation, the SOEmn bit of serial output enable register m (SOEm) is cleared to 0 before receiving the last data.

**Figure 12-114. Flowchart of Stop Condition Generation**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), r: IIC number (r = 00, 01, 10, 11, 20, 21), mn = 00 to 03, 10, 11

### 12.8.5 Calculating transfer rate

The transfer rate for simplified $I^2C$ (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication can be calculated by the following expressions.

(Transfer rate) = {Operation clock ($f_{MCK}$) frequency of target channel} ÷ (SDRmn[15:9] + 1) ÷ 2

**Caution SDRmn[15:9] must not be set to 00000000B. Be sure to set a value of 00000001B or greater for SDRmn[15:9]. The duty ratio of the SCL signal output by the simplified $I^2C$ is 50%. The $I^2C$ bus specifications define that the low-level width of the SCL signal is longer than the high-level width. If 400 kbps (fast mode) or 1 Mbps (fast mode plus) is specified, therefore, the low-level width of the SCL output signal becomes shorter than the value specified in the $I^2C$ bus specifications. Make sure that the SDRmn[15:9] value satisfies the $I^2C$ bus specifications.**

**Remarks 1.** The value of SDRmn[15:9] is the value of bits 15 to 9 of the SDRmn register (0000001B to 1111111B) and therefore is 1 to 127.

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

The operation clock ($f_{MCK}$) is determined by serial clock select register m (SPSm) and bit 15 (CKSmn) of serial mode register mn (SMRmn).

**Table 12-5. Selection of Operation Clock For Simplified I²C**

| SMRmn Register | SPSm Register | | | | | | | | Operation Clock ($f_{MCK}$)[Note] | |
|---|---|---|---|---|---|---|---|---|---|---|
| CKSmn | PRS m13 | PRS m12 | PRS m11 | PRS m10 | PRS m03 | PRS m02 | PRS m01 | PRS m00 | | $f_{CLK}$ = 32 MHz |
| 0 | X | X | X | X | 0 | 0 | 0 | 0 | $f_{CLK}$ | 32 MHz |
| | X | X | X | X | 0 | 0 | 0 | 1 | $f_{CLK}/2$ | 16 MHz |
| | X | X | X | X | 0 | 0 | 1 | 0 | $f_{CLK}/2^2$ | 8 MHz |
| | X | X | X | X | 0 | 0 | 1 | 1 | $f_{CLK}/2^3$ | 4 MHz |
| | X | X | X | X | 0 | 1 | 0 | 0 | $f_{CLK}/2^4$ | 2 MHz |
| | X | X | X | X | 0 | 1 | 0 | 1 | $f_{CLK}/2^5$ | 1 MHz |
| | X | X | X | X | 0 | 1 | 1 | 0 | $f_{CLK}/2^6$ | 500 kHz |
| | X | X | X | X | 0 | 1 | 1 | 1 | $f_{CLK}/2^7$ | 250 kHz |
| | X | X | X | X | 1 | 0 | 0 | 0 | $f_{CLK}/2^8$ | 125 kHz |
| | X | X | X | X | 1 | 0 | 0 | 1 | $f_{CLK}/2^9$ | 62.5 kHz |
| | X | X | X | X | 1 | 0 | 1 | 0 | $f_{CLK}/2^{10}$ | 31.25 kHz |
| | X | X | X | X | 1 | 0 | 1 | 1 | $f_{CLK}/2^{11}$ | 15.63 kHz |
| 1 | 0 | 0 | 0 | 0 | X | X | X | X | $f_{CLK}$ | 32 MHz |
| | 0 | 0 | 0 | 1 | X | X | X | X | $f_{CLK}/2$ | 16 MHz |
| | 0 | 0 | 1 | 0 | X | X | X | X | $f_{CLK}/2^2$ | 8 MHz |
| | 0 | 0 | 1 | 1 | X | X | X | X | $f_{CLK}/2^3$ | 4 MHz |
| | 0 | 1 | 0 | 0 | X | X | X | X | $f_{CLK}/2^4$ | 2 MHz |
| | 0 | 1 | 0 | 1 | X | X | X | X | $f_{CLK}/2^5$ | 1 MHz |
| | 0 | 1 | 1 | 0 | X | X | X | X | $f_{CLK}/2^6$ | 500 kHz |
| | 0 | 1 | 1 | 1 | X | X | X | X | $f_{CLK}/2^7$ | 250 kHz |
| | 1 | 0 | 0 | 0 | X | X | X | X | $f_{CLK}/2^8$ | 125 kHz |
| | 1 | 0 | 0 | 1 | X | X | X | X | $f_{CLK}/2^9$ | 62.5 kHz |
| | 1 | 0 | 1 | 0 | X | X | X | X | $f_{CLK}/2^{10}$ | 31.25 kHz |
| | 1 | 0 | 1 | 1 | X | X | X | X | $f_{CLK}/2^{11}$ | 15.63 kHz |
| Other than above | | | | | | | | | Setting prohibited | |

**Note** When changing the clock selected for $f_{CLK}$ (by changing the system clock control register (CKC) value), do so after having stopped (serial channel stop register m (STm) = 000FH) the operation of the serial array unit (SAU).

**Remarks 1.** X: Don't care

**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), mn = 00 to 03, 10, 11

Here is an example of setting an I²C transfer rate where $f_{MCK} = f_{CLK}$ = 32 MHz.

| I²C Transfer Mode (Desired Transfer Rate) | $f_{CLK}$ = 32 MHz | | | |
|---|---|---|---|---|
| | Operation Clock ($f_{MCK}$) | SDRmn[15:9] | Calculated Transfer Rate | Error from Desired Transfer Rate |
| 100 kHz | $f_{CLK}/2$ | 79 | 100 kHz | 0.0% |
| 400 kHz | $f_{CLK}$ | 41 | 380 kHz | 5.0%[Note] |
| 1 MHz | $f_{CLK}$ | 18 | 0.84 MHz | 16.0%[Note] |

**Note** The error cannot be set to about 0% because the duty ratio of the SCL signal is 50%.

### 12.8.6 Procedure for processing errors that occurred during simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication

The procedure for processing errors that occurred during simplified I²C (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21) communication is described in Figure 12-115 and 12-116.

**Figure 12-115. Processing Procedure in Case of Overrun Error**

| Software Manipulation | Hardware Status | Remark |
|---|---|---|
| Reads serial data register mn (SDRmn). → | The BFFmn bit of the SSRmn register is set to 0 and channel n is enabled to receive data. | This is to prevent an overrun error if the next reception is completed during error processing. |
| Reads serial status register mn (SSRmn). | | Error type is identified and the read value is used to clear error flag. |
| Writes 1 to serial flag clear trigger register mn (SIRmn). → | Error flag is cleared. | Error can be cleared only during reading, by writing the value read from the SSRmn register to the SIRmn register without modification. |

<R>

**Figure 12-116. Processing Procedure in Case of ACK Error in Simplified I²C Mode**

| Software Manipulation | Hardware Status | Remark |
|---|---|---|
| Reads serial status register mn (SSRmn). | | Error type is identified and the read value is used to clear error flag. |
| Writes serial flag clear trigger register mn (SIRmn). → | Error flag is cleared. | Error can be cleared only during reading, by writing the value read from the SSRmn register to the SIRmn register without modification. |
| Sets the STmn bit of serial channel stop register m (STm) to 1. → | The SEmn bit of serial channel enable status register m (SEm) is set to 0 and channel n stops operation. | Slave is not ready for reception because ACK is not returned. Therefore, a stop condition is created, the bus is released, and communication is started again from the start condition. Or, a restart condition is generated and transmission can be redone from address transmission. |
| Creates stop condition. | | |
| Creates start condition. | | |
| Sets the SSmn bit of serial channel start register m (SSm) to 1. → | The SEmn bit of serial channel enable status register m (SEm) is set to 1 and channel n is enabled to operate. | |

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), r: IIC number (r = 00, 01, 10, 11, 20, 21)
mn = 00 to 03, 10, 11

# CHAPTER 13 SERIAL INTERFACE IICA

## 13.1 Functions of Serial Interface IICA

Serial interface IICA has the following three modes.

**(1) Operation stop mode**

This mode is used when serial transfers are not performed. It can therefore be used to reduce power consumption.

**(2) $I^2C$ bus mode (multimaster supported)**

This mode is used for 8-bit data transfers with several devices via two lines: a serial clock (SCLA0) line and a serial data bus (SDAA0) line.

This mode complies with the $I^2C$ bus format and the master device can generated “start condition”, “address”, “transfer direction specification”, “data”, and “stop condition” data to the slave device, via the serial data bus. The slave device automatically detects these received status and data by hardware. This function can simplify the part of application program that controls the $I^2C$ bus.

Since the SCLA0 and SDAA0 pins are used for open drain outputs, serial interface IICA requires pull-up resistors for the serial clock line and the serial data bus line.

**(3) Wakeup mode**

The STOP mode can be released by generating an interrupt request signal (INTIICA0) when an extension code from the master device or a local address has been received while in STOP mode. This can be set by using the WUP0 bit of IICA control register 01 (IICCTL01).

Figure 13-1 shows a block diagram of serial interface IICA.

**Figure 13-1. Block Diagram of Serial Interface IICA**

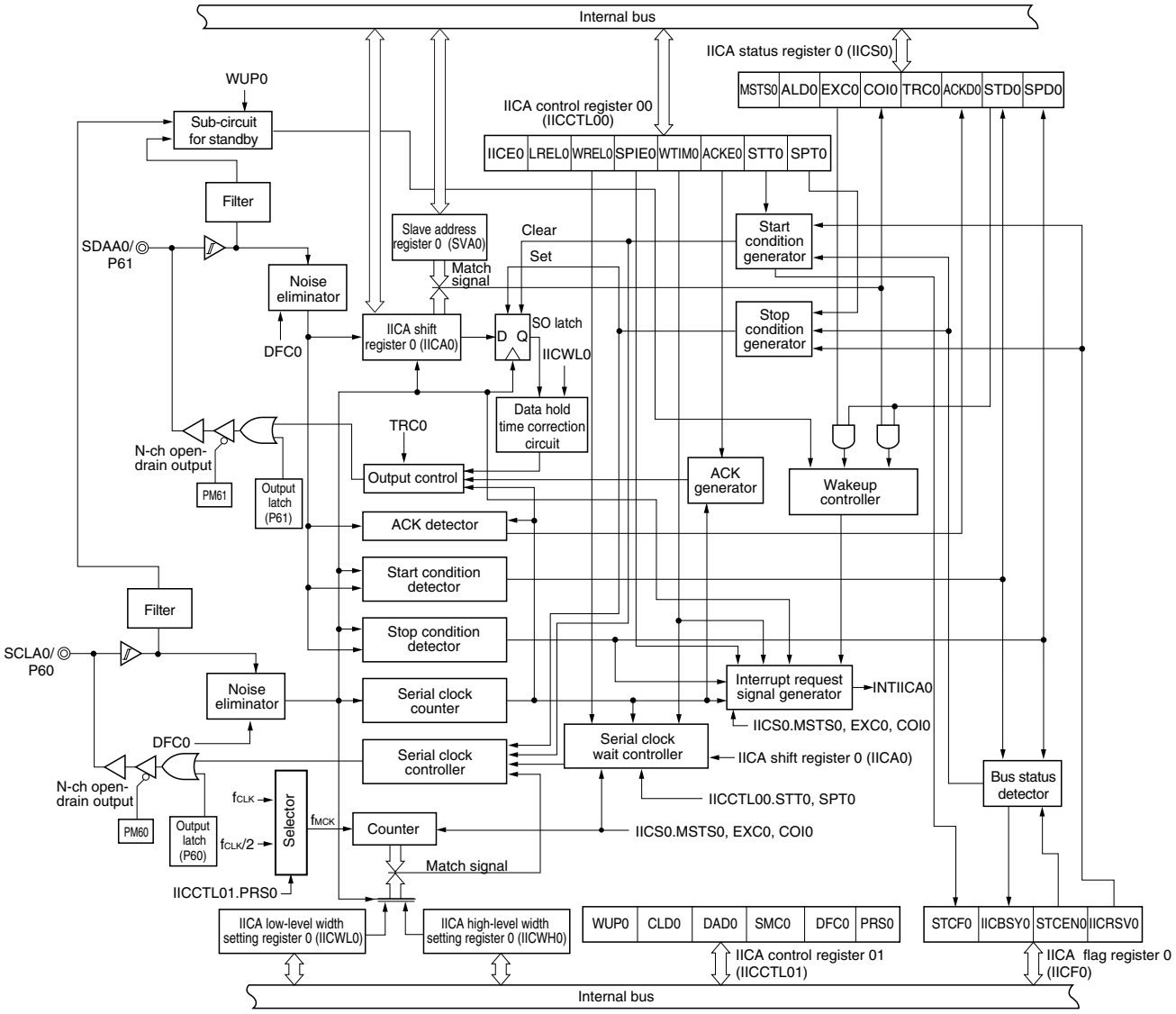
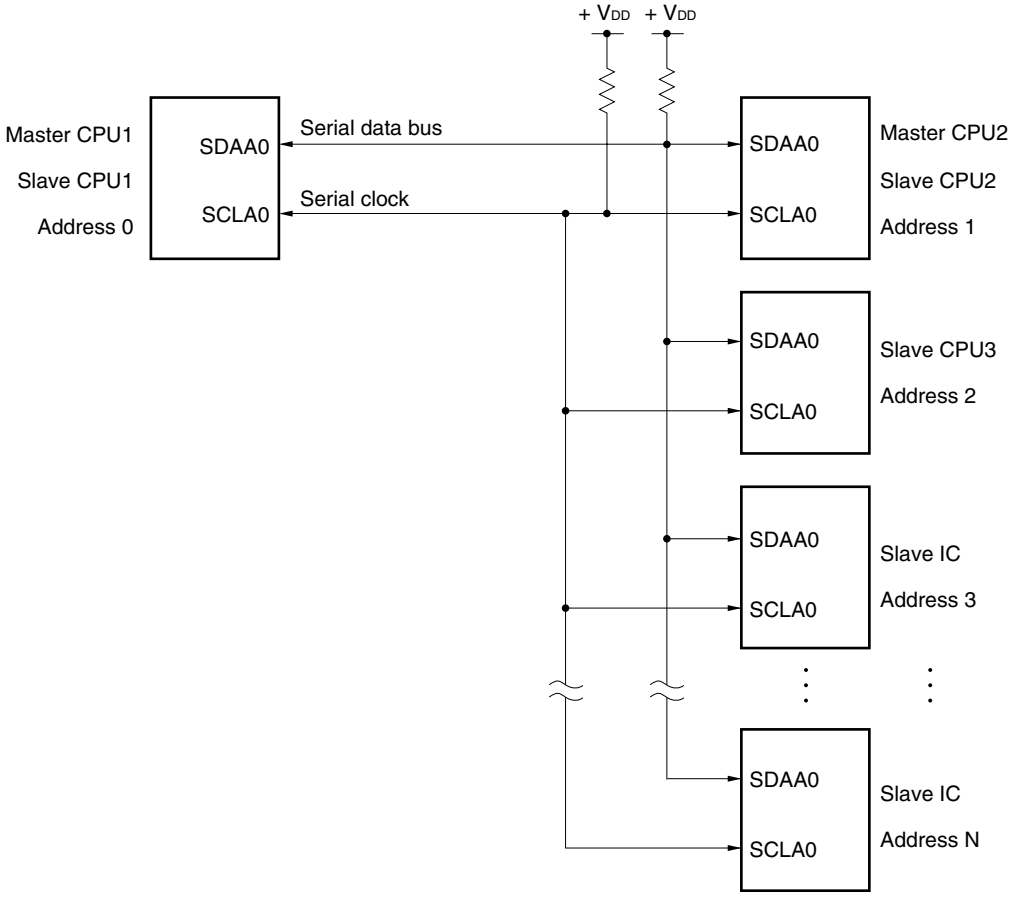


Figure 13-2 shows a serial bus configuration example.

**Figure 13-2. Serial Bus Configuration Example Using I²C Bus**



## 13.2 Configuration of Serial Interface IICA

Serial interface IICA includes the following hardware.

**Table 13-1. Configuration of Serial Interface IICA**

| Item | Configuration |
|---|---|
| Registers | IICA shift register 0 (IICA0)<br>Slave address register 0 (SVA0) |
| Control registers | Peripheral enable register 0 (PER0)<br>IICA control register 00 (IICCTL00)<br>IICA status register 0 (IICS0)<br>IICA flag register 0 (IICF0)<br>IICA control register 01 (IICCTL01)<br>IICA low-level width setting register 0 (IICWL0)<br>IICA high-level width setting register 0 (IICWH0)<br>Port mode register 6 (PM6)<br>Port register 6 (P6) |

### (1) IICA shift register 0 (IICA0)

The IICA0 register is used to convert 8-bit serial data to 8-bit parallel data and vice versa in synchronization with the serial clock. The IICA0 register can be used for both transmission and reception.

The actual transmit and receive operations can be controlled by writing and reading operations to the IICA0 register.

Cancel the wait state and start data transfer by writing data to the IICA0 register during the wait period.

The IICA0 register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears IICA0 to 00H.

**Figure 13-3. Format of IICA Shift Register 0 (IICA0)**

Address: FFF50H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IICA0 | | | | | | | | |

**Cautions 1. Do not write data to the IICA0 register during data transfer.**

**2. Write or read the IICA0 register only during the wait period. Accessing the IICA0 register in a communication state other than during the wait period is prohibited. When the device serves as the master, however, the IICA0 register can be written only once after the communication trigger bit (STT0) is set to 1.**

**3. When communication is reserved, write data to the IICA0 register after the interrupt triggered by a stop condition is detected.**

### (2) Slave address register 0 (SVA0)

This register stores seven bits of local addresses {A6, A5, A4, A3, A2, A1, A0} when in slave mode.

The SVA0 register can be set by an 8-bit memory manipulation instruction.

However, rewriting to this register is prohibited while STD0 = 1 (while the start condition is detected).

Reset signal generation clears the SVA0 register to 00H.

**Figure 13-4. Format of Slave Address Register 0 (SVA0)**

Address: F0234H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| SVA0 | A6 | A5 | A4 | A3 | A2 | A1 | A0 | 0Note |

**Note** Bit 0 is fixed to 0.

**(3) SO latch**

The SO latch is used to retain the SDAA0 pin's output level.

**(4) Wakeup controller**

This circuit generates an interrupt request (INTIICA0) when the address received by this register matches the address value set to the slave address register 0 (SVA0) or when an extension code is received.

**(5) Serial clock counter**

This counter counts the serial clocks that are output or input during transmit/receive operations and is used to verify that 8-bit data was transmitted or received.

**(6) Interrupt request signal generator**

This circuit controls the generation of interrupt request signals (INTIICA0).

An $I^2C$ interrupt request is generated by the following two triggers.

- Falling edge of eighth or ninth clock of the serial clock (set by the WTIM0 bit)
- Interrupt request generated when a stop condition is detected (set by the SPIE0 bit)

**Remark** WTIM0 bit: Bit 3 of IICA control register 00 (IICCTL00)
SPIE0 bit: Bit 4 of IICA control register 00 (IICCTL00)

**(7) Serial clock controller**

In master mode, this circuit generates the clock output via the SCLA0 pin from a sampling clock.

**(8) Serial clock wait controller**

This circuit controls the wait timing.

**(9) ACK generator, stop condition detector, start condition detector, and ACK detector**

These circuits generate and detect each status.

**(10) Data hold time correction circuit**

This circuit generates the hold time for data corresponding to the falling edge of the serial clock.

**(11) Start condition generator**

This circuit generates a start condition when the STT0 bit is set to 1.

However, in the communication reservation disabled status (IICRSV bit = 1), when the bus is not released (IICBSY bit = 1), start condition requests are ignored and the STCF bit is set to 1.

**(12) Stop condition generator**

This circuit generates a stop condition when the SPT0 bit is set to 1.

**(13) Bus status detector**

This circuit detects whether or not the bus is released by detecting start conditions and stop conditions.

However, as the bus status cannot be detected immediately following operation, the initial status is set by the STCEN bit.

**Remark** STT0 bit: Bit 1 of IICA control register 00 (IICCTL00)
SPT0 bit: Bit 0 of IICA control register 00 (IICCTL00)
IICRSV bit: Bit 0 of IICA flag register 0 (IICF0)
IICBSY bit: Bit 6 of IICA flag register 0 (IICF0)
STCF bit: Bit 7 of IICA flag register 0 (IICF0)
STCEN bit: Bit 1 of IICA flag register 0 (IICF0)

## 13.3 Registers Controlling Serial Interface IICA

Serial interface IICA0 is controlled by the following registers.

- Peripheral enable register 0 (PER0)
- IICA control register 00 (IICCTL00)
- IICA flag register 0 (IICF0)
- IICA status register 0 (IICS0)
- IICA control register 01 (IICCTL01)
- IICA low-level width setting register 0 (IICWL0)
- IICA high-level width setting register 0 (IICWH0)
- Port mode register 6 (PM6)
- Port register 6 (P6)

### 13.3.1 Peripheral enable register 0 (PER0)

This register is used to enable or disable supplying the clock to the peripheral hardware. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

When serial interface IICA0 is used, be sure to set bit 4 (IICA0EN) of this register to 1.

The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 13-5. Format of Peripheral Enable Register 0 (PER0)**

Address: F00F0H After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PER0 | RTCEN | 0 | ADCEN | IICA0EN | SAU1EN[Note] | SAU0EN | 0 | TAU0EN |

| IICA0EN | Control of serial interface IICA0 input clock supply |
|---|---|
| 0 | Stops input clock supply.<br>• SFR used by serial interface IICA0 cannot be written.<br>• Serial interface IICA0 is in the reset status. |
| 1 | Enables input clock supply.<br>• SFR used by serial interface IICA0 can be read/written. |

**Note** 32-, 48-, and 64-pin products only.

**Cautions 1. When setting serial interface IICA0, be sure to set the following registers first while the IICA0EN bit is set to 1. If IICA0EN = 0, the control registers of serial interface IICA are set to their initial values, and writing to them is ignored (except for port mode register 6 (PM6) and port register 6 (P6)).**
**• IICA control register 00 (IICCTL00)**
**• IICA flag register 0 (IICF0)**
**• IICA status register 0 (IICS0)**
**• IICA control register 01 (IICCTL01)**
**• IICA low-level width setting register 0 (IICWL0)**
**• IICA high-level width setting register 0 (IICWH0)**

**2. Be sure to clear the following bits to 0.**
**25-pin products: bits 1, 3, 6**
**32-, 48-, 64-pin products: bits 1, 6**

### 13.3.2 IICA control register 00 (IICCTL00)

This register is used to enable/stop $I^2C$ operations, set wait timing, and set other $I^2C$ operations.

The IICCTL00 register can be set by a 1-bit or 8-bit memory manipulation instruction. However, set the SPIE0, WTIM0, and ACKE0 bits while IICE0 = 0 or during the wait period. These bits can be set at the same time when the IICE0 bit is set from “0” to “1”.

Reset signal generation clears this register to 00H.

**Figure 13-6. Format of IICA Control Register 00 (IICCTL00) (1/4)**

Address: F0230H After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| IICCTL00 | IICE0 | LREL0 | WREL0 | SPIE0 | WTIM0 | ACKE0 | STT0 | SPT0 |

| IICE0 | $I^2C$ operation enable |
|---|---|
| 0 | Stop operation. Reset the IICA status register 0 (IICS0)[Note 1]. Stop internal operation. |
| 1 | Enable operation. |
| Be sure to set this bit (1) while the SCLA0 and SDAA0 lines are at high level. | |
| Condition for clearing (IICE0 = 0) | Condition for setting (IICE0 = 1) |
| • Cleared by instruction<br>• Reset | • Set by instruction |

| LREL0[Notes 2, 3] | Exit from communications |
|---|---|
| 0 | Normal operation |
| 1 | This exits from the current communications and sets standby mode. This setting is automatically cleared to 0 after being executed.<br>Its uses include cases in which a locally irrelevant extension code has been received.<br>The SCLA0 and SDAA0 lines are set to high impedance.<br>The following flags of IICA control register 00 (IICCTL00) and the IICA status register 0 (IICS0) are cleared to 0.<br>• STT0 • SPT0 • MSTS0 • EXC0 • COI0 • TRC0 • ACKD0 • STD0 |
| The standby mode following exit from communications remains in effect until the following communications entry conditions are met.<br>• After a stop condition is detected, restart is in master mode.<br>• An address match or extension code reception occurs after the start condition. | |
| Condition for clearing (LREL0 = 0) | Condition for setting (LREL0 = 1) |
| • Automatically cleared after execution<br>• Reset | • Set by instruction |

| WREL0[Notes 2, 3] | Wait cancellation |
|---|---|
| 0 | Do not cancel wait |
| 1 | Cancel wait. This setting is automatically cleared after wait is canceled. |
| When the WREL0 bit is set (wait canceled) during the wait period at the ninth clock pulse in the transmission status (TRC0 = 1), the SDAA0 line goes into the high impedance state (TRC0 = 0). | |
| Condition for clearing (WREL0 = 0) | Condition for setting (WREL0 = 1) |
| • Automatically cleared after execution<br>• Reset | • Set by instruction |

**Notes 1.** The IICA status register 0 (IICS0), the STCF and IICBSY bits of the IICA flag register 0 (IICF0), and the CLD0 and DAD0 bits of IICA control register 01 (IICCTL01) are reset.

**2.** The signal of this bit is invalid while IICE0 is 0.

**3.** When the LREL0 and WREL0 bits are read, 0 is always read.

**Caution If the operation of $I^2C$ is enabled (IICE0 = 1) when the SCLA0 line is high level, the SDAA0 line is low level, and the digital filter is turned on (DFC0 bit of IICCTL01 register = 1), a start condition will be inadvertently detected immediately. In this case, set (1) the LREL0 bit by using a 1-bit memory manipulation instruction immediately after enabling operation of $I^2C$ (IICE0 = 1).**

**Figure 13-6. Format of IICA Control Register 00 (IICCTL00) (2/4)**

| SPIE0[Note 1] | Enable/disable generation of interrupt request when stop condition is detected |
|---|---|
| 0 | Disable |
| 1 | Enable |
| If the WUP0 bit of IICA control register 01 (IICCTL01) is 1, no stop condition interrupt will be generated even if SPIE0 = 1. | |
| Condition for clearing (SPIE0 = 0) | Condition for setting (SPIE0 = 1) |
| • Cleared by instruction<br>• Reset | • Set by instruction |

| WTIM0[Note 1] | Control of wait and interrupt request generation |
|---|---|
| 0 | Interrupt request is generated at the eighth clock's falling edge.<br>Master mode: After output of eight clocks, clock output is set to low level and wait is set.<br>Slave mode: After input of eight clocks, the clock is set to low level and wait is set for master device. |
| 1 | Interrupt request is generated at the ninth clock's falling edge.<br>Master mode: After output of nine clocks, clock output is set to low level and wait is set.<br>Slave mode: After input of nine clocks, the clock is set to low level and wait is set for master device. |
| An interrupt is generated at the falling edge of the ninth clock during address transfer independently of the setting of this bit. The setting of this bit is valid when the address transfer is completed. When in master mode, a wait is inserted at the falling edge of the ninth clock during address transfers. For a slave device that has received a local address, a wait is inserted at the falling edge of the ninth clock after an acknowledge ($\overline{ACK}$) is issued. However, when the slave device has received an extension code, a wait is inserted at the falling edge of the eighth clock. | |
| Condition for clearing (WTIM0 = 0) | Condition for setting (WTIM0 = 1) |
| • Cleared by instruction<br>• Reset | • Set by instruction |

| ACKE0[Notes 1, 2] | Acknowledgment control |
|---|---|
| 0 | Disable acknowledgment. |
| 1 | Enable acknowledgment. During the ninth clock period, the SDAA0 line is set to low level. |
| Condition for clearing (ACKE0 = 0) | Condition for setting (ACKE0 = 1) |
| • Cleared by instruction<br>• Reset | • Set by instruction |

**Notes 1.** The signal of this bit is invalid while IICE0 is 0. Set this bit during that period.

**2.** The set value is invalid during address transfer and if the code is not an extension code.
When the device serves as a slave and the addresses match, an acknowledgment is generated regardless of the set value.

**Figure 13-6. Format of IICA Control Register 00 (IICCTL00) (3/4)**

| STT0[Notes 1, 2] | Start condition trigger |
|---|---|
| 0 | Do not generate a start condition. |
| 1 | When bus is released (in standby state, when IICBSY = 0):<br>If this bit is set (1), a start condition is generated (startup as the master).<br>When a third party is communicating:<br>• When communication reservation function is enabled (IICRSV = 0)<br>Functions as the start condition reservation flag. When set to 1, automatically generates a start condition after the bus is released.<br>• When communication reservation function is disabled (IICRSV = 1)<br>Even if this bit is set (1), the STT0 bit is cleared and the STT0 clear flag (STCF) is set (1). No start condition is generated.<br>In the wait state (when master device):<br>Generates a restart condition after releasing the wait. |

Cautions concerning set timing
- For master reception: Cannot be set to 1 during transfer. Can be set to 1 only in the waiting period when the ACKE0 bit has been cleared to 0 and slave has been notified of final reception.
- For master transmission: A start condition cannot be generated normally during the acknowledge period. Set to 1 during the wait period that follows output of the ninth clock.
- Cannot be set to 1 at the same time as stop condition trigger (SPT0).
- Once STT0 is set (1), setting it again (1) before the clear condition is met is not allowed.

| Condition for clearing (STT0 = 0) | Condition for setting (STT0 = 1) |
|---|---|
| • Cleared by setting the STT0 bit to 1 while communication reservation is prohibited.<br>• Cleared by loss in arbitration<br>• Cleared after start condition is generated by master device<br>• Cleared by LREL0 = 1 (exit from communications)<br>• When IICE0 = 0 (operation stop)<br>• Reset | • Set by instruction |

**Notes 1.** The signal of this bit is invalid while IICE0 is 0.

<R> **2.** The STT0 bit is always read as 0.

**Remark** IICRSV: Bit 0 of IIC flag register 0 (IICF0)
STCF: Bit 7 of IIC flag register 0 (IICF0)

**Figure 13-6. Format of IICA Control Register 00 (IICCTL00) (4/4)**

| SPT0[Note] | Stop condition trigger |
|---|---|
| 0 | Stop condition is not generated. |
| 1 | Stop condition is generated (termination of master device's transfer). |

Cautions concerning set timing
- For master reception: Cannot be set to 1 during transfer. Can be set to 1 only in the waiting period when the ACKE0 bit has been cleared to 0 and slave has been notified of final reception.
- For master transmission: A stop condition cannot be generated normally during the acknowledge period. Therefore, set it during the wait period that follows output of the ninth clock.
- Cannot be set to 1 at the same time as start condition trigger (STT0).
- The SPT0 bit can be set to 1 only when in master mode.
- When the WTIM0 bit has been cleared to 0, if the SPT0 bit is set to 1 during the wait period that follows output of eight clocks, note that a stop condition will be generated during the high-level period of the ninth clock. The WTIM0 bit should be changed from 0 to 1 during the wait period following the output of eight clocks, and the SPT0 bit should be set to 1 during the wait period that follows the output of the ninth clock.
- Once STT0 is set (1), setting it again (1) before the clear condition is met is not allowed.

| Condition for clearing (SPT0 = 0) | Condition for setting (SPT0 = 1) |
|---|---|
| • Cleared by loss in arbitration<br>• Automatically cleared after stop condition is detected<br>• Cleared by LREL0 = 1 (exit from communications)<br>• When IICE0 = 0 (operation stop)<br>• Reset | • Set by instruction |

<R> **Note** The SPT0 bit is always read as 0.

**Caution When bit 3 (TRC0) of the IICA status register 0 (IICS0) is set to 1 (transmission status), bit 5 (WREL0) of IICA control register 00 (IICCTL00) is set to 1 during the ninth clock and wait is canceled, after which the TRC0 bit is cleared (reception status) and the SDAA0 line is set to high impedance. Release the wait performed while the TRC0 bit is 1 (transmission status) by writing to the IICA shift register 0.**

**Remark** Bit 0 (SPT0) becomes 0 when it is read after data setting.

### 13.3.3 IICA status register 0 (IICS0)

This register indicates the status of $I^2C$.

The IICS0 register is read by a 1-bit or 8-bit memory manipulation instruction only when STT0 = 1 and during the wait period.

Reset signal generation clears this register to 00H.

**Caution Reading the IICS0 register while the address match wakeup function is enabled (WUP0 = 1) in STOP mode is prohibited. When the WUP0 bit is changed from 1 to 0 (wakeup operation is stopped), regardless of the INTIICA0 interrupt request, the change in status is not reflected until the next start condition or stop condition is detected. To use the wakeup function, therefore, enable (SPIE0 = 1) the interrupt generated by detecting a stop condition and read the IICS0 register after the interrupt has been detected.**

**Remark** STT0: bit 1 of IICA control register 00 (IICCTL00)
WUP0: bit 7 of IICA control register 01 (IICCTL01)

**Figure 13-7. Format of IICA Status Register 0 (IICS0) (1/3)**

Address: FFF51H After reset: 00H R

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| IICS0 | MSTS0 | ALD0 | EXC0 | COI0 | TRC0 | ACKD0 | STD0 | SPD0 |

| MSTS0 | Master status check flag |
|---|---|
| 0 | Slave device status or communication standby status |
| 1 | Master device communication status |

| Condition for clearing (MSTS0 = 0) | Condition for setting (MSTS0 = 1) |
|---|---|
| • When a stop condition is detected<br>• When ALD0 = 1 (arbitration loss)<br>• Cleared by LREL0 = 1 (exit from communications)<br>• When the IICE0 bit changes from 1 to 0 (operation stop)<br>• Reset | • When a start condition is generated |

| ALD0 | Detection of arbitration loss |
|---|---|
| 0 | This status means either that there was no arbitration or that the arbitration result was a “win”. |
| 1 | This status indicates the arbitration result was a “loss”. The MSTS0 bit is cleared. |

| Condition for clearing (ALD0 = 0) | Condition for setting (ALD0 = 1) |
|---|---|
| • Automatically cleared after the IICS0 register is read[Note]<br>• When the IICE0 bit changes from 1 to 0 (operation stop)<br>• Reset | • When the arbitration result is a “loss”. |

**Note** This register is also cleared when a 1-bit memory manipulation instruction is executed for bits other than the IICS0 register. Therefore, when using the ALD0 bit, read the data of this bit before the data of the other bits.

**Remark** LREL0: Bit 6 of IICA control register 00 (IICCTL00)
IICE0: Bit 7 of IICA control register 00 (IICCTL00)

**Figure 13-7. Format of IICA Status Register 0 (IICS0) (2/3)**

| EXC0 | Detection of extension code reception |
|---|---|
| 0 | Extension code was not received. |
| 1 | Extension code was received. |

| Condition for clearing (EXC0 = 0) | Condition for setting (EXC0 = 1) |
|---|---|
| • When a start condition is detected<br>• When a stop condition is detected<br>• Cleared by LREL0 = 1 (exit from communications)<br>• When the IICE0 bit changes from 1 to 0 (operation stop)<br>• Reset | • When the higher four bits of the received address data is either “0000” or “1111” (set at the rising edge of the eighth clock). |

| COI0 | Detection of matching addresses |
|---|---|
| 0 | Addresses do not match. |
| 1 | Addresses match. |

| Condition for clearing (COI0 = 0) | Condition for setting (COI0 = 1) |
|---|---|
| • When a start condition is detected<br>• When a stop condition is detected<br>• Cleared by LREL0 = 1 (exit from communications)<br>• When the IICE0 bit changes from 1 to 0 (operation stop)<br>• Reset | • When the received address matches the local address (slave address register 0 (SVA0)) (set at the rising edge of the eighth clock). |

| TRC0 | Detection of transmit/receive status |
|---|---|
| 0 | Receive status (other than transmit status). The SDAA0 line is set for high impedance. |
| 1 | Transmit status. The value in the SO0 latch is enabled for output to the SDAA0 line (valid starting at the falling edge of the first byte’s ninth clock). |

| Condition for clearing (TRC0 = 0) | Condition for setting (TRC0 = 1) |
|---|---|
| <Both master and slave><br>• When a stop condition is detected<br>• Cleared by LREL0 = 1 (exit from communications)<br>• When the IICE0 bit changes from 1 to 0 (operation stop)<br>• Cleared by WREL0 = 1[Note] (wait cancel)<br>• When the ALD0 bit changes from 0 to 1 (arbitration loss)<br>• Reset<br>• When not used for communication (MSTS0, EXC0, COI0 = 0)<br><Master><br>• When “1” is output to the first byte’s LSB (transfer direction specification bit)<br><Slave><br>• When a start condition is detected<br>• When “0” is input to the first byte’s LSB (transfer direction specification bit) | <Master><br>• When a start condition is generated<br>• When 0 (master transmission) is output to the LSB (transfer direction specification bit) of the first byte (during address transfer)<br><Slave><br>• When 1 (slave transmission) is input to the LSB (transfer direction specification bit) of the first byte from the master (during address transfer) |

**Note** When bit 3 (TRC0) of the IICA status register 0 (IICS0) is set to 1 (transmission status), bit 5 (WREL0) of IICA control register 00 (IICCTL00) is set to 1 during the ninth clock and wait is canceled, after which the TRC0 bit is cleared (reception status) and the SDAA0 line is set to high impedance. Release the wait performed while the TRC0 bit is 1 (transmission status) by writing to the IICA shift register 0.

**Remark** LREL0: Bit 6 of IICA control register 00 (IICCTL00)
IICE0: Bit 7 of IICA control register 00 (IICCTL00)

**Figure 13-7. Format of IICA Status Register 0 (IICS0) (3/3)**

| ACKD0 | Detection of acknowledge ($\overline{ACK}$) |
|---|---|
| 0 | Acknowledge was not detected. |
| 1 | Acknowledge was detected. |

| Condition for clearing (ACKD0 = 0) | Condition for setting (ACKD0 = 1) |
|---|---|
| • When a stop condition is detected<br>• At the rising edge of the next byte's first clock<br>• Cleared by LREL0 = 1 (exit from communications)<br>• When the IICE0 bit changes from 1 to 0 (operation stop)<br>• Reset | • After the SDAA0 line is set to low level at the rising edge of SCLA0 line's ninth clock |

| STD0 | Detection of start condition |
|---|---|
| 0 | Start condition was not detected. |
| 1 | Start condition was detected. This indicates that the address transfer period is in effect. |

| Condition for clearing (STD0 = 0) | Condition for setting (STD0 = 1) |
|---|---|
| • When a stop condition is detected<br>• At the rising edge of the next byte's first clock following address transfer<br>• Cleared by LREL0 = 1 (exit from communications)<br>• When the IICE0 bit changes from 1 to 0 (operation stop)<br>• Reset | • When a start condition is detected |

| SPD0 | Detection of stop condition |
|---|---|
| 0 | Stop condition was not detected. |
| 1 | Stop condition was detected. The master device's communication is terminated and the bus is released. |

| Condition for clearing (SPD0 = 0) | Condition for setting (SPD0 = 1) |
|---|---|
| • At the rising edge of the address transfer byte's first clock following setting of this bit and detection of a start condition<br>• When the WUP0 bit changes from 1 to 0<br>• When the IICE0 bit changes from 1 to 0 (operation stop)<br>• Reset | • When a stop condition is detected |

**Remark** LREL0: Bit 6 of IICA control register 00 (IICCTL00)
IICE0: Bit 7 of IICA control register 00 (IICCTL00)

### 13.3.4 IICA flag register 0 (IICF0)

This register sets the operation mode of $I^2C$ and indicates the status of the $I^2C$ bus.

The IICF0 register can be set by a 1-bit or 8-bit memory manipulation instruction. However, the STT0 clear flag (STCF) and $I^2C$ bus status flag (IICBSY) bits are read-only.

The IICRSV bit can be used to enable/disable the communication reservation function.

The STCEN bit can be used to set the initial value of the IICBSY bit.

The IICRSV and STCEN bits can be written only when the operation of $I^2C$ is disabled (bit 7 (IICE0) of IICA control register 00 (IICCTL00) = 0). When operation is enabled, the IICF0 register can be read.

Reset signal generation clears this register to 00H.

**Figure 13-8. Format of IICA Flag Register 0 (IICF0)**

Address: FFF52H After reset: 00H R/W[Note]

| Symbol | <7> | <6> | 5 | 4 | 3 | 2 | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| IICF0 | STCF0 | IICBSY0 | 0 | 0 | 0 | 0 | STCEN0 | IICRSV0 |

| STCF0 | STT0 clear flag |
|---|---|
| 0 | Generate start condition |
| 1 | Start condition generation unsuccessful: clear the STT0 flag |

| Condition for clearing (STCF0 = 0) | Condition for setting (STCF0 = 1) |
|---|---|
| • Cleared by STT0 = 1<br>• When IICE0 = 0 (operation stop)<br>• Reset | • Generating start condition unsuccessful and the STT0 bit cleared to 0 when communication reservation is disabled (IICRSV0 = 1). |

| IICBSY0 | $I^2C$ bus status flag |
|---|---|
| 0 | Bus release status (communication initial status when STCEN0 = 1) |
| 1 | Bus communication status (communication initial status when STCEN0 = 0) |

| Condition for clearing (IICBSY0 = 0) | Condition for setting (IICBSY0 = 1) |
|---|---|
| • Detection of stop condition<br>• When IICE0 = 0 (operation stop)<br>• Reset | • Detection of start condition<br>• Setting of the IICE0 bit when STCEN0 = 0 |

| STCEN0 | Initial start enable trigger |
|---|---|
| 0 | After operation is enabled (IICE0 = 1), enable generation of a start condition upon detection of a stop condition. |
| 1 | After operation is enabled (IICE0 = 1), enable generation of a start condition without detecting a stop condition. |

| Condition for clearing (STCEN0 = 0) | Condition for setting (STCEN0 = 1) |
|---|---|
| • Cleared by instruction<br>• Detection of start condition<br>• Reset | • Set by instruction |

| IICRSV0 | Communication reservation function disable bit |
|---|---|
| 0 | Enable communication reservation |
| 1 | Disable communication reservation |

| Condition for clearing (IICRSV0 = 0) | Condition for setting (IICRSV0 = 1) |
|---|---|
| • Cleared by instruction<br>• Reset | • Set by instruction |

**Note** Bits 6 and 7 are read-only.

**Cautions 1. Write to the STCEN bit only when the operation is stopped (IICE0 = 0).**

**2. As the bus release status (IICBSY = 0) is recognized regardless of the actual bus status when STCEN = 1, when generating the first start condition (STT0 = 1), it is necessary to verify that no third party communications are in progress in order to prevent such communications from being destroyed.**

**3. Write to IICRSV only when the operation is stopped (IICE0 = 0).**

**Remark** STT0: Bit 1 of IICA control register 00 (IICCTL00)
IICE0: Bit 7 of IICA control register 00 (IICCTL00)

### 13.3.5 IICA control register 01 (IICCTL01)

This register is used to set the operation mode of $I^2C$ and detect the statuses of the SCLA0 and SDAA0 pins.

The IICCTL01 register can be set by a 1-bit or 8-bit memory manipulation instruction. However, the CLD0 and DAD0 bits are read-only.

Set the IICCTL01 register, except the WUP0 bit, while operation of $I^2C$ is disabled (bit 7 (IICE0) of IICA control register 00 (IICCTL00) is 0).

Reset signal generation clears this register to 00H.

**Figure 13-9. Format of IICA Control Register 01 (IICCTL01) (1/2)**

Address: F0231H After reset: 00H R/W[Note 1]

| Symbol | <7> | 6 | <5> | <4> | <3> | <2> | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| IICCTL01 | WUP0 | 0 | CLD0 | DAD0 | SMC0 | DFC0 | 0 | PRS0 |

| WUP0 | Control of address match wakeup |
|---|---|
| 0 | Stops operation of address match wakeup function in STOP mode. |
| 1 | Enables operation of address match wakeup function in STOP mode. |

To shift to STOP mode when WUP0 = 1, execute the STOP instruction at least three clocks of $f_{MCK}$ after setting (1) the WUP0 bit (see **Figure 13-22 Flow When Setting WUP0 = 1**).
Clear (0) the WUP0 bit after the address has matched or an extension code has been received. The subsequent communication can be entered by the clearing (0) WUP0 bit. (The wait must be released and transmit data must be written after the WUP0 bit has been cleared (0).)
The interrupt timing when the address has matched or when an extension code has been received, while WUP0 = 1, is identical to the interrupt timing when WUP0 = 0. (A delay of the difference of sampling by the clock will occur.) Furthermore, when WUP0 = 1, a stop condition interrupt is not generated even if the SPIE0 bit is set to 1.
When WUP0 = 0 is set by a source other than an interrupt from serial interface IICA, operation as the master device cannot be performed until the subsequent start condition or stop condition is detected. Do not output a start condition by setting (1) the STT0 bit, without waiting for the detection of the subsequent start condition or stop condition.

| Condition for clearing (WUP0 = 0) | Condition for setting (WUP0 = 1) |
|---|---|
| • Cleared by instruction (after address match or extension code reception) | • Set by instruction (when the MSTS0, EXC0, and COI0 bits are “0”, and the STD0 bit also “0” (communication not entered))[Note 2] |

**Notes 1.** Bits 4 and 5 are read-only.

**2.** The status of the IICA status register 0 (IICS0) must be checked and the WUP0 bit must be set during the period shown below.

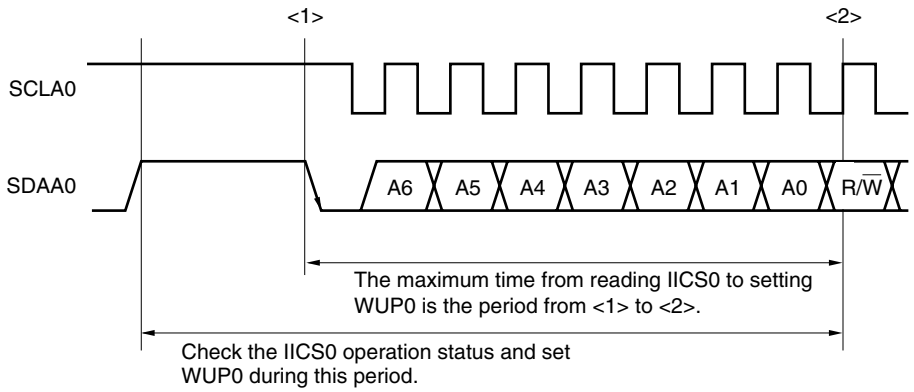


**Figure 13-9. Format of IICA Control Register 01 (IICCTL01) (2/2)**

| CLD0 | Detection of SCLA0 pin level (valid only when IICE0 = 1) |
|---|---|
| 0 | The SCLA0 pin was detected at low level. |
| 1 | The SCLA0 pin was detected at high level. |

| Condition for clearing (CLD0 = 0) | Condition for setting (CLD0 = 1) |
|---|---|
| • When the SCLA0 pin is at low level<br>• When IICE0 = 0 (operation stop)<br>• Reset | • When the SCLA0 pin is at high level |

| DAD0 | Detection of SDAA0 pin level (valid only when IICE0 = 1) |
|---|---|
| 0 | The SDAA0 pin was detected at low level. |
| 1 | The SDAA0 pin was detected at high level. |

| Condition for clearing (DAD0 = 0) | Condition for setting (DAD0 = 1) |
|---|---|
| • When the SDAA0 pin is at low level<br>• When IICE0 = 0 (operation stop)<br>• Reset | • When the SDAA0 pin is at high level |

| SMC0 | Operation mode switching |
|---|---|
| 0 | Operates in standard mode (fastest transfer rate: 100 kbps). |
| 1 | Operates in fast mode (fastest transfer rate: 400 kbps) or fast mode plus (fastest transfer rate: 1 Mbps). |

| DFC0 | Digital filter operation control |
|---|---|
| 0 | Digital filter off. |
| 1 | Digital filter on. |
| Use the digital filter only in fast mode and fast mode plus.<br>The digital filter is used for noise elimination.<br>The transfer clock does not vary, regardless of the DFC0 bit being set (1) or cleared (0). | |

| PRS0 | Control of IICA operation clock ($f_{MCK}$) |
|---|---|
| 0 | Selects $f_{CLK}$ (1 MHz ≤ $f_{CLK}$ ≤ 20 MHz). |
| 1 | Selects $f_{CLK}/2$ (20 MHz < $f_{CLK}$). |

<R> **Cautions 1. The fastest operation frequency of the IICA operation clock ($f_{MCK}$) is 20 MHz (Max.). Set the bit 0 (PRS0) of the IICA control register 01 (IICCTL01) to 1, only when the $f_{CLK}$ exceeds 20 MHz.**

<R> **2. Note the minimum $f_{CLK}$ operation frequency when setting the transfer clock.**
**The minimum $f_{CLK}$ operation frequency for serial interface IICA is determined according to the mode.**
**Fast mode: $f_{CLK}$ = 3.5 MHz (MIN.)**
**Fast mode plus: $f_{CLK}$ = 10 MHz (MIN.)**
**Normal mode: $f_{CLK}$ = 1 MHz (MIN.)**

**Remark** IICE0: Bit 7 of IICA control register 00 (IICCTL00)

### 13.3.6 IICA low-level width setting register 0 (IICWL0)

This register is used to control the low-level width ($t_{LOW}$) of the SCLA0 pin signal that is output by serial interface IICA and the SDAA0 pin signal.

The IICWL0 register can be set by an 8-bit memory manipulation instruction.

Set the IICWL0 register while operation of $I^2C$ is disabled (bit 7 (IICE0) of IICA control register 00 (IICCTL00) is 0).

Reset signal generation sets this register to FFH.

For details about setting the IICWL0 register, see **13.4.2 Setting transfer clock by using IICWL0 and IICWH0 registers**.

**Figure 13-10. Format of IICA Low-Level Width Setting Register 0 (IICWL0)**

Address: F0232H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IICWL0 | | | | | | | | |

### 13.3.7 IICA high-level width setting register 0 (IICWH0)

This register is used to control the high-level width of the SCLA0 pin signal that is output by serial interface IICA and the SDAA0 pin signal.

The IICWH0 register can be set by an 8-bit memory manipulation instruction.

Set the IICWH0 register while operation of $I^2C$ is disabled (bit 7 (IICE0) of IICA control register 00 (IICCTL00) is 0).

Reset signal generation sets this register to FFH.

**Figure 13-11. Format of IICA High-Level Width Setting Register 0 (IICWH0)**

Address: F0233H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IICWH0 | | | | | | | | |

<R> **Remark** For setting procedures of the transfer clock on master side and of the IICWL0 and IICWH0 registers on slave side, see **13.4.2 (1)** and **13.4.2 (2)**, respectively.

### 13.3.8 Port mode register 6 (PM6)

This register sets the input/output of port 6 in 1-bit units.

When using the P60/SCLA0 pin as clock I/O and the P61/SDAA0 pin as serial data I/O, clear PM60 and PM61, and the output latches of P60 and P61 to 0.

Set the IICE0 bit (bit 7 of IICA control register 00 (IICCTL00)) to 1 before setting the output mode because the P60/SCLA0 and P61/SDAA0 pins output a low level (fixed) when the IICE0 bit is 0.

The PM6 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to FFH.

**Figure 13-12. Format of Port Mode Register 6 (PM6) (in Case of 64-pin Products)**

Address: FFF26H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM6 | 1 | 1 | 1 | 1 | PM63 | PM62 | PM61 | PM60 |

| PM6n | PM6n pin I/O mode selection (n = 0 to 3) |
|---|---|
| 0 | Output mode (output buffer on) |
| 1 | Input mode (output buffer off) |

## 13.4 $I^2C$ Bus Mode Functions

### 13.4.1 Pin configuration

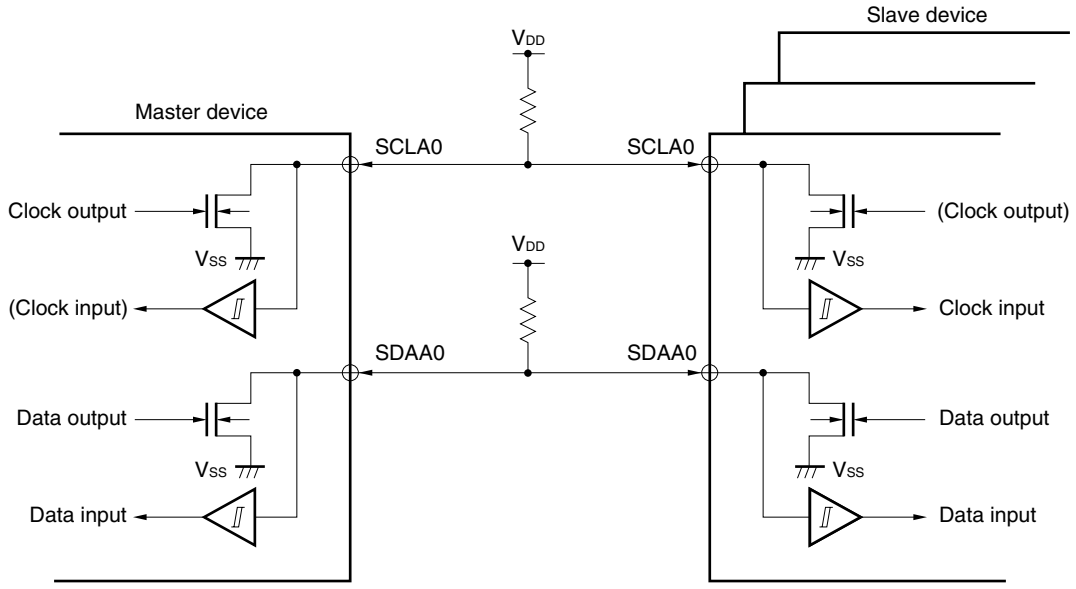
The serial clock pin (SCLA0) and the serial data bus pin (SDAA0) are configured as follows.

(1) SCLA0 .... This pin is used for serial clock input and output.
This pin is an N-ch open-drain output for both master and slave devices. Input is Schmitt input.

(2) SDAA0 .... This pin is used for serial data input and output.
This pin is an N-ch open-drain output for both master and slave devices. Input is Schmitt input.

Since outputs from the serial clock line and the serial data bus line are N-ch open-drain outputs, an external pull-up resistor is required.

**Figure 13-13. Pin Configuration Diagram**



### 13.4.2 Setting transfer clock by using IICWL0 and IICWH0 registers

#### (1) Setting transfer clock on master side

$$\text{Transfer clock} = \frac{f_{MCK}}{IICWL0 + IICWH0 + f_{MCK}(t_R + t_F)}$$

At this time, the optimal setting values of the IICWL0 and IICWH0 registers are as follows.
(The fractional parts of all setting values are rounded up.)

- When the fast mode

$$IICWL0 = \frac{0.52}{\text{Transfer clock}} \times f_{MCK}$$
$$IICWH0 = (\frac{0.48}{\text{Transfer clock}} - t_R - t_F) \times f_{MCK}$$

- When the normal mode

$$IICWL0 = \frac{0.47}{\text{Transfer clock}} \times f_{MCK}$$
$$IICWH0 = (\frac{0.53}{\text{Transfer clock}} - t_R - t_F) \times f_{MCK}$$

- When the fast mode plus

$$IICWL0 = \frac{0.50}{\text{Transfer clock}} \times f_{MCK}$$
$$IICWH0 = (\frac{0.50}{\text{Transfer clock}} - t_R - t_F) \times f_{MCK}$$

#### (2) Setting IICWL0 and IICWH0 registers on slave side

(The fractional parts of all setting values are truncated.)

- When the fast mode

$$IICWL0 = 1.3\ \mu s \times f_{MCK}$$
$$IICWH0 = (1.2\ \mu s - t_R - t_F) \times f_{MCK}$$

- When the normal mode

$$IICWL0 = 4.7\ \mu s \times f_{MCK}$$
$$IICWH0 = (5.3\ \mu s - t_R - t_F) \times f_{MCK}$$

- When the fast mode plus

$$IICWL0 = 0.50\ \mu s \times f_{MCK}$$
$$IICWH0 = (0.50\ \mu s - t_R - t_F) \times f_{MCK}$$

(**Caution** and **Remarks** are listed on the next page.)

**Cautions 1. The fastest operation frequency of the IICA operation clock ($f_{MCK}$) is 20 MHz (Max.).**
**Set the bit 0 (PRS0) of the IICA control register 01 (IICCTL01) to 1, only when the $f_{CLK}$ exceeds 20 MHz.**

**2. Note the minimum $f_{CLK}$ operation frequency when setting the transfer clock.**
**The minimum $f_{CLK}$ operation frequency for serial interface IICA is determined according to the mode.**
**Fast mode: $f_{CLK}$ = 3.5 MHz (MIN.)**
**Fast mode plus: $f_{CLK}$ = 10 MHz (MIN.)**
**Normal mode: $f_{CLK}$ = 1 MHz (MIN.)**

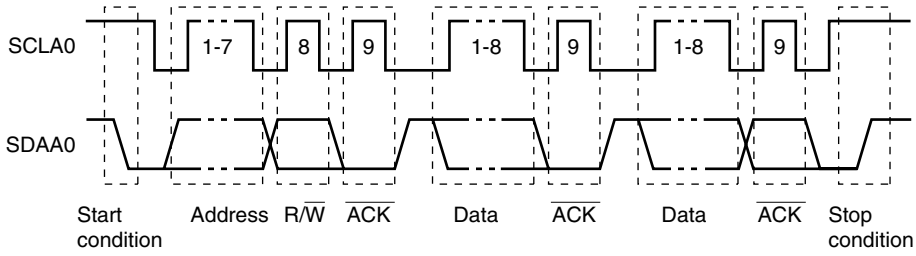
**Remarks 1.** Calculate the rise time ($t_R$) and fall time ($t_F$) of the SDAA0 and SCLA0 signals separately, because they differ depending on the pull-up resistance and wire load.

**2.** IICWL0: IICA low-level width setting register 0
IICWH0: IICA high-level width setting register 0
$t_F$: SDAA0 and SCLA0 signal falling times
$t_R$: SDAA0 and SCLA0 signal rising times
<R> $f_{MCK}$: IICA operation clock frequency

## 13.5 I²C Bus Definitions and Control Methods

The following section describes the I²C bus's serial data communication format and the signals used by the I²C bus. Figure 13-14 shows the transfer timing for the "start condition", "address", "data", and "stop condition" output via the I²C bus's serial data bus.

**Figure 13-14. I²C Bus Serial Data Transfer Timing**



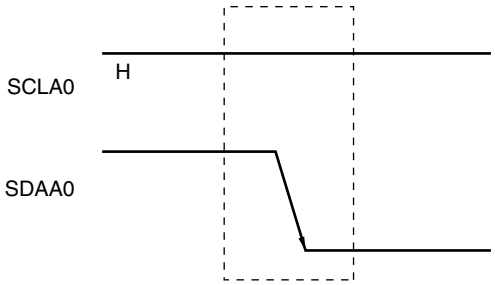

The master device generates the start condition, slave address, and stop condition.
The acknowledge ($\overline{ACK}$) can be generated by either the master or slave device (normally, it is output by the device that receives 8-bit data).

The serial clock (SCLA0) is continuously output by the master device. However, in the slave device, the SCLA0 pin low level period can be extended and a wait can be inserted.

### 13.5.1 Start conditions

A start condition is met when the SCLA0 pin is at high level and the SDAA0 pin changes from high level to low level. The start conditions for the SCLA0 pin and SDAA0 pin are signals that the master device generates to the slave device when starting a serial transfer. When the device is used as a slave, start conditions can be detected.

**Figure 13-15. Start Conditions**




A start condition is output when bit 1 (STT0) of IICA control register 00 (IICCTL00) is set (1) after a stop condition has been detected (SPD0: Bit 0 of the IICA status register 0 (IICS0) = 1). When a start condition is detected, bit 1 (STD0) of the IICS0 register is set (1).

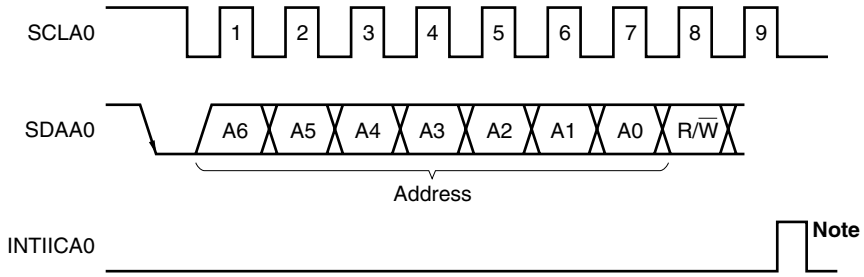
### 13.5.2 Addresses

The address is defined by the 7 bits of data that follow the start condition.

An address is a 7-bit data segment that is output in order to select one of the slave devices that are connected to the master device via the bus lines. Therefore, each slave device connected via the bus lines must have a unique address.

The slave devices include hardware that detects the start condition and checks whether or not the 7-bit address data matches the data values stored in the slave address register 0 (SVA0). If the address data matches the SVA0 register values, the slave device is selected and communicates with the master device until the master device generates a start condition or stop condition.

**Figure 13-16. Address**




**Note** INTIICA0 is not issued if data other than a local address or extension code is received during slave device operation.

Addresses are output when a total of 8 bits consisting of the slave address and the transfer direction described in **13.5.3 Transfer direction specification** are written to the IICA shift register 0 (IICA0). The received addresses are written to the IICA0 register.

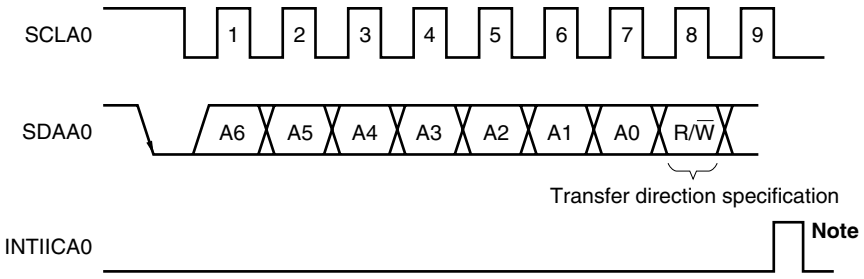
The slave address is assigned to the higher 7 bits of the IICA0 register.

### 13.5.3 Transfer direction specification

In addition to the 7-bit address data, the master device sends 1 bit that specifies the transfer direction.

When this transfer direction specification bit has a value of “0”, it indicates that the master device is transmitting data to a slave device. When the transfer direction specification bit has a value of “1”, it indicates that the master device is receiving data from a slave device.

**Figure 13-17. Transfer Direction Specification**




**Note** INTIICA0 is not issued if data other than a local address or extension code is received during slave device operation.

### 13.5.4 Acknowledge ($\overline{ACK}$)

$\overline{ACK}$ is used to check the status of serial data at the transmission and reception sides.

The reception side returns $\overline{ACK}$ each time it has received 8-bit data.

The transmission side usually receives $\overline{ACK}$ after transmitting 8-bit data. When $\overline{ACK}$ is returned from the reception side, it is assumed that reception has been correctly performed and processing is continued. Whether $\overline{ACK}$ has been detected can be checked by using bit 2 (ACKD0) of the IICA status register 0 (IICS0).

When the master receives the last data item, it does not return $\overline{ACK}$ and instead generates a stop condition. If a slave does not return $\overline{ACK}$ after receiving data, the master outputs a stop condition or restart condition and stops transmission. If $\overline{ACK}$ is not returned, the possible causes are as follows.

<1> Reception was not performed normally.
<2> The final data item was received.
<3> The reception side specified by the address does not exist.

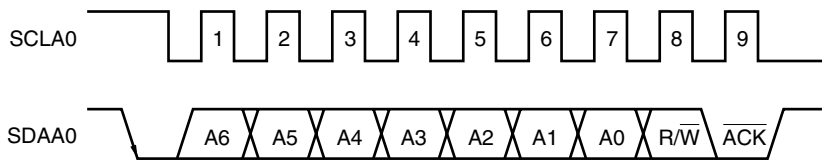
To generate $\overline{ACK}$, the reception side makes the SDAA0 line low at the ninth clock (indicating normal reception).

Automatic generation of $\overline{ACK}$ is enabled by setting bit 2 (ACKE0) of IICA control register 00 (IICCTL00) to 1. Bit 3 (TRC0) of the IICS0 register is set by the data of the eighth bit that follows 7-bit address information. Usually, set the ACKE0 bit to 1 for reception (TRC0 = 0).

If a slave can receive no more data during reception (TRC0 = 0) or does not require the next data item, then the slave must inform the master, by clearing the ACKE0 bit to 0, that it will not receive any more data.

When the master does not require the next data item during reception (TRC0 = 0), it must clear the ACKE0 bit to 0 so that $\overline{ACK}$ is not generated. In this way, the master informs a slave at the transmission side that it does not require any more data (transmission will be stopped).

**Figure 13-18. $\overline{ACK}$**




When the local address is received, $\overline{ACK}$ is automatically generated, regardless of the value of the ACKE0 bit. When an address other than that of the local address is received, $\overline{ACK}$ is not generated (NACK).

When an extension code is received, $\overline{ACK}$ is generated if the ACKE0 bit is set to 1 in advance.

How $\overline{ACK}$ is generated when data is received differs as follows depending on the setting of the wait timing.

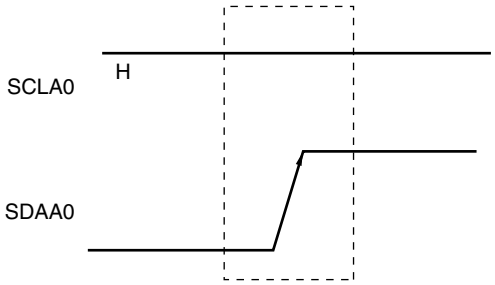
- When 8-clock wait state is selected (bit 3 (WTIM0) of IICCTL00 register = 0):
  By setting the ACKE0 bit to 1 before releasing the wait state, $\overline{ACK}$ is generated at the falling edge of the eighth clock of the SCLA0 pin.
- When 9-clock wait state is selected (bit 3 (WTIM0) of IICCTL00 register = 1):
  $\overline{ACK}$ is generated by setting the ACKE0 bit to 1 in advance.

### 13.5.5 Stop condition

When the SCLA0 pin is at high level, changing the SDAA0 pin from low level to high level generates a stop condition.

A stop condition is a signal that the master device generates to the slave device when serial transfer has been completed. When the device is used as a slave, stop conditions can be detected.

**Figure 13-19. Stop Condition**




A stop condition is generated when bit 0 (SPT0) of IICA control register 00 (IICCTL00) is set to 1. When the stop condition is detected, bit 0 (SPD0) of the IICA status register 0 (IICS0) is set to 1 and INTIICA0 is generated when bit 4 (SPIE0) of the IICCTL00 register is set to 1.

### 13.5.6 Wait

The wait is used to notify the communication partner that a device (master or slave) is preparing to transmit or receive data (i.e., is in a wait state).

Setting the SCLA0 pin to low level notifies the communication partner of the wait state. When wait state has been canceled for both the master and slave devices, the next data transfer can begin.

**Figure 13-20. Wait (1/2)**

**(1) When master device has a nine-clock wait and slave device has an eight-clock wait (master transmits, slave receives, and ACKE0 = 1)**

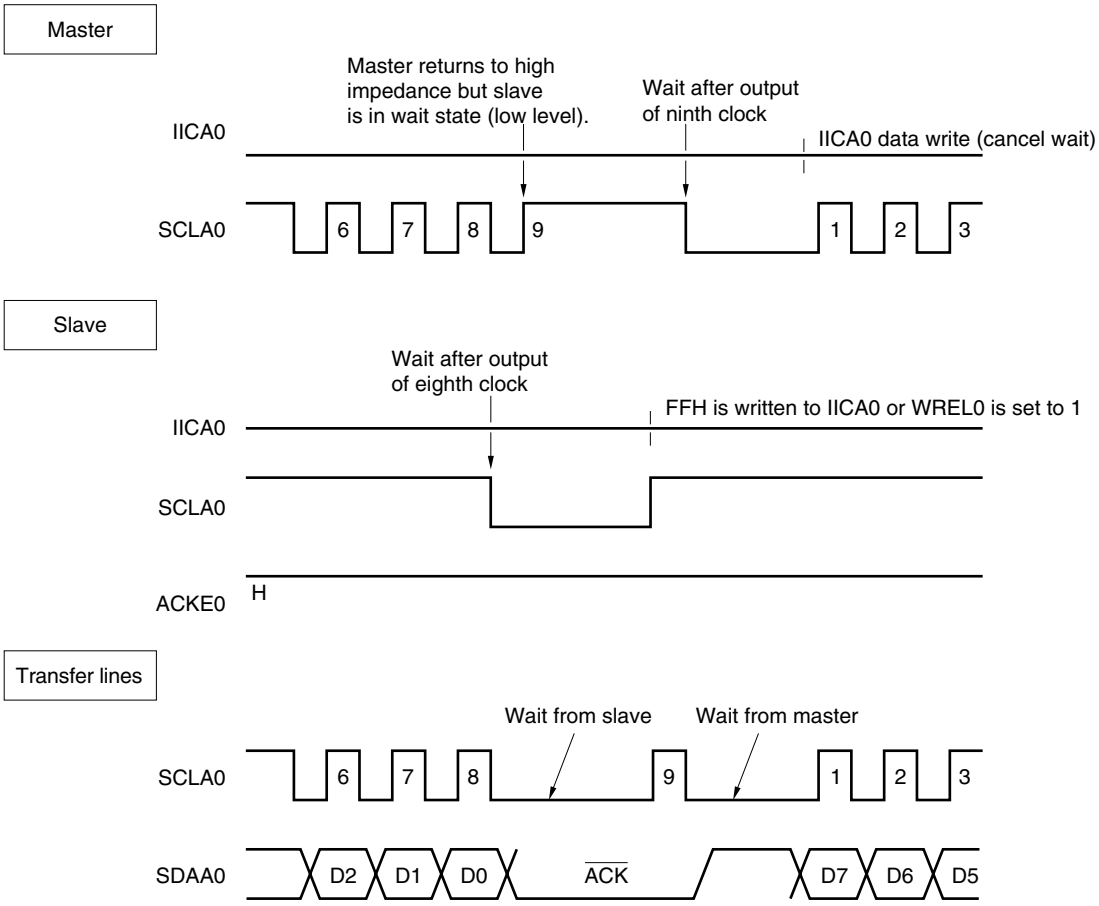
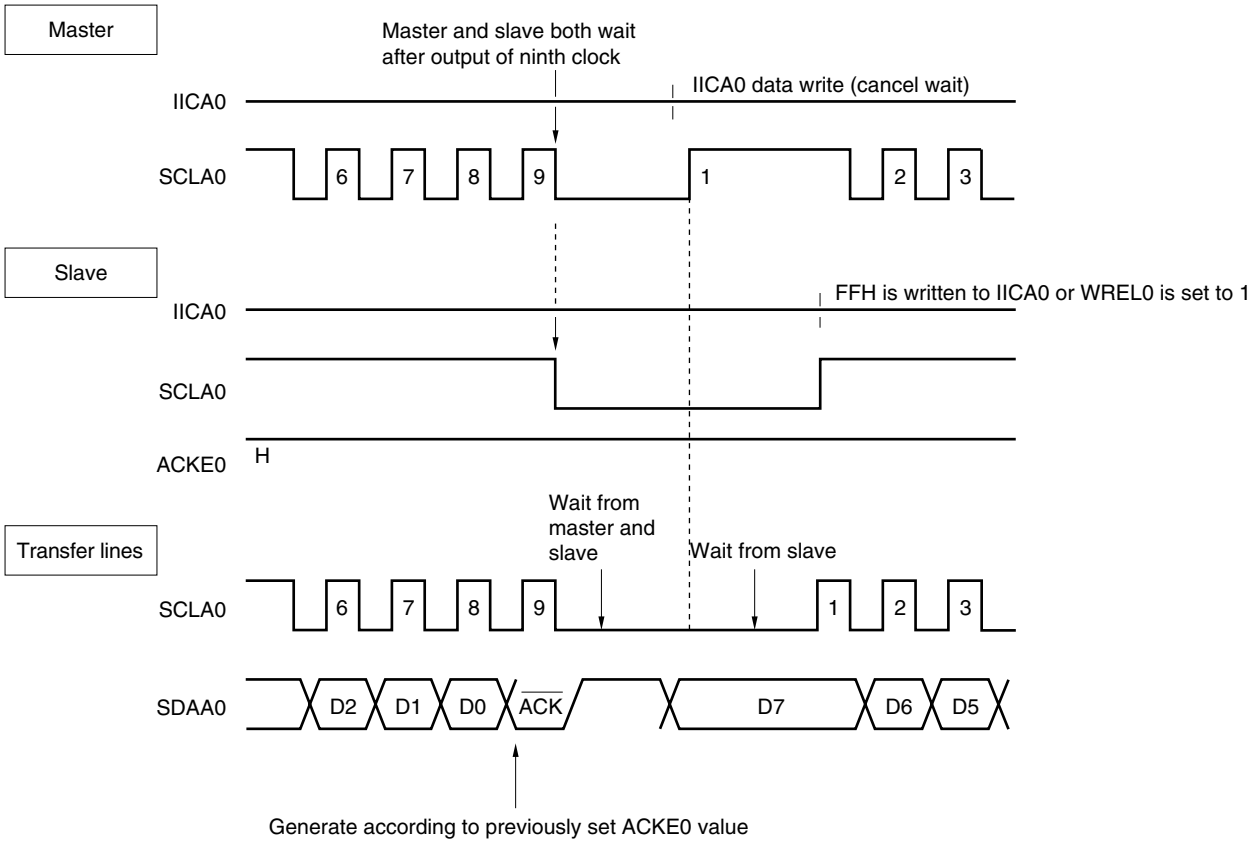


**Figure 13-20. Wait (2/2)**

**(2) When master and slave devices both have a nine-clock wait (master transmits, slave receives, and ACKE0 = 1)**




**Remark** ACKE0: Bit 2 of IICA control register 00 (IICCTL00)
WREL0: Bit 5 of IICA control register 00 (IICCTL00)

A wait may be automatically generated depending on the setting of bit 3 (WTIM0) of IICA control register 00 (IICCTL00).

Normally, the receiving side cancels the wait state when bit 5 (WREL0) of the IICCTL00 register is set to 1 or when FFH is written to the IICA shift register 0 (IICA0), and the transmitting side cancels the wait state when data is written to the IICA0 register.

The master device can also cancel the wait state via either of the following methods.

- By setting bit 1 (STT0) of the IICCTL00 register to 1
- By setting bit 0 (SPT0) of the IICCTL00 register to 1

### 13.5.7 Canceling wait

The $I^2C$ usually cancels a wait state by the following processing.

- Writing data to the IICA shift register 0 (IICA0)
- Setting bit 5 (WREL0) of IICA control register 00 (IICCTL00) (canceling wait)
- Setting bit 1 (STT0) of the IICCTL00 register (generating start condition)[Note]
- Setting bit 0 (SPT0) of the IICCTL00 register (generating stop condition)[Note]

**Note** Master only

When the above wait canceling processing is executed, the $I^2C$ cancels the wait state and communication is resumed.

To cancel a wait state and transmit data (including addresses), write the data to the IICA0 register.

To receive data after canceling a wait state, or to complete data transmission, set bit 5 (WREL0) of the IICCTL00 register to 1.

To generate a restart condition after canceling a wait state, set bit 1 (STT0) of the IICCTL00 register to 1.

To generate a stop condition after canceling a wait state, set bit 0 (SPT0) of the IICCTL00 register to 1.

Execute the canceling processing only once for one wait state.

If, for example, data is written to the IICA0 register after canceling a wait state by setting the WREL0 bit to 1, an incorrect value may be output to SDAA0 line because the timing for changing the SDAA0 line conflicts with the timing for writing the IICA0 register.

In addition to the above, communication is stopped if the IICE0 bit is cleared to 0 when communication has been aborted, so that the wait state can be canceled.

If the $I^2C$ bus has deadlocked due to noise, processing is saved from communication by setting bit 6 (LREL0) of the IICCTL00 register, so that the wait state can be canceled.

**Caution If a processing to cancel a wait state is executed when WUP0 = 1, the wait state will not be canceled.**

### 13.5.8 Interrupt request (INTIICA0) generation timing and wait control

The setting of bit 3 (WTIM0) of IICA control register 00 (IICCTL00) determines the timing by which INTIICA0 is generated and the corresponding wait control, as shown in Table 13-2.

**Table 13-2. INTIICA0 Generation Timing and Wait Control**

| WTIM0 | During Slave Device Operation | | | During Master Device Operation | | |
|---|---|---|---|---|---|---|
| | Address | Data Reception | Data Transmission | Address | Data Reception | Data Transmission |
| 0 | 9 Notes 1, 2 | 8 Note 2 | 8 Note 2 | 9 | 8 | 8 |
| 1 | 9 Notes 1, 2 | 9 Note 2 | 9 Note 2 | 9 | 9 | 9 |

**Notes 1.** The slave device's INTIICA0 signal and wait period occurs at the falling edge of the ninth clock only when there is a match with the address set to the slave address register 0 (SVA0).
At this point, $\overline{ACK}$ is generated regardless of the value set to the IICCTL00 register's bit 2 (ACKE0). For a slave device that has received an extension code, INTIICA0 occurs at the falling edge of the eighth clock.
However, if the address does not match after restart, INTIICA0 is generated at the falling edge of the 9th clock, but wait does not occur.

**2.** If the received address does not match the contents of the slave address register 0 (SVA0) and extension code is not received, neither INTIICA0 nor a wait occurs.

**Remark** The numbers in the table indicate the number of the serial clock's clock signals. Interrupt requests and wait control are both synchronized with the falling edge of these clock signals.

**(1) During address transmission/reception**

- Slave device operation: Interrupt and wait timing are determined depending on the conditions described in Notes 1 and 2 above, regardless of the WTIM0 bit.
- Master device operation: Interrupt and wait timing occur at the falling edge of the ninth clock regardless of the WTIM0 bit.

**(2) During data reception**

- Master/slave device operation: Interrupt and wait timing are determined according to the WTIM0 bit.

**(3) During data transmission**

- Master/slave device operation: Interrupt and wait timing are determined according to the WTIM0 bit.

**(4) Wait cancellation method**

The four wait cancellation methods are as follows.

- Writing data to the IICA shift register 0 (IICA0)
- Setting bit 5 (WREL0) of IICA control register 00 (IICCTL00) (canceling wait)
- Setting bit 1 (STT0) of IICCTL00 register (generating start condition) Note
- Setting bit 0 (SPT0) of IICCTL00 register (generating stop condition) Note

**Note** Master only.

When an 8-clock wait has been selected (WTIM0 = 0), the presence/absence of $\overline{ACK}$ generation must be determined prior to wait cancellation.

**(5) Stop condition detection**

INTIICA0 is generated when a stop condition is detected (only when SPIE0 = 1).

### 13.5.9 Address match detection method

In $I^2C$ bus mode, the master device can select a particular slave device by transmitting the corresponding slave address.

Address match can be detected automatically by hardware. An interrupt request (INTIICA0) occurs when the address set to the slave address register 0 (SVA0) matches the slave address sent by the master device, or when an extension code has been received.

### 13.5.10 Error detection

In $I^2C$ bus mode, the status of the serial data bus (SDAA0) during data transmission is captured by the IICA shift register 0 (IICA0) of the transmitting device, so the IICA data prior to transmission can be compared with the transmitted IICA data to enable detection of transmission errors. A transmission error is judged as having occurred when the compared data values do not match.

### 13.5.11 Extension code

(1) When the higher 4 bits of the receive address are either “0000” or “1111”, the extension code reception flag (EXC0) is set to 1 for extension code reception and an interrupt request (INTIICA0) is issued at the falling edge of the eighth clock. The local address stored in the slave address register 0 (SVA0) is not affected.

(2) The settings below are specified if 11110xx0 is transferred from the master by using a 10-bit address transfer when the SVA0 register is set to 11110xx0. Note that INTIICA0 occurs at the falling edge of the eighth clock.

- Higher four bits of data match: EXC0 = 1
- Seven bits of data match: COI0 = 1

**Remark** EXC0: Bit 5 of IICA status register 0 (IICS0)
COI0: Bit 4 of IICA status register 0 (IICS0)

(3) Since the processing after the interrupt request occurs differs according to the data that follows the extension code, such processing is performed by software.
If the extension code is received while a slave device is operating, then the slave device is participating in communication even if its address does not match.
For example, after the extension code is received, if you do not wish to operate the target device as a slave device, set bit 6 (LREL0) of IICA control register 00 (IICCTL00) to 1 to set the standby mode for the next communication operation.

**Table 13-3. Bit Definitions of Major Extension Codes**

| Slave Address | R/W Bit | Description |
|---|---|---|
| 0 0 0 0 0 0 0 | 0 | General call address |
| 1 1 1 1 0 x x | 0 | 10-bit slave address specification (during address authentication) |
| 1 1 1 1 0 x x | 1 | 10-bit slave address specification (after address match, when read command is issued) |

**Remark** See the $I^2C$ bus specifications issued by NXP Semiconductors for details of extension codes other than those described above.

### 13.5.12 Arbitration

When several master devices simultaneously generate a start condition (when the STT0 bit is set to 1 before the STD0 bit is set to 1), communication among the master devices is performed as the number of clocks are adjusted until the data differs. This kind of operation is called arbitration.

When one of the master devices loses in arbitration, an arbitration loss flag (ALD0) in the IICA status register 0 (IICS0) is set (1) via the timing by which the arbitration loss occurred, and the SCLA0 and SDAA0 lines are both set to high impedance, which releases the bus.

The arbitration loss is detected based on the timing of the next interrupt request (the eighth or ninth clock, when a stop condition is detected, etc.) and the ALD0 = 1 setting that has been made by software.

For details of interrupt request timing, see **13.5.8 Interrupt request (INTIICA0) generation timing and wait control**.

**Remark** STD0: Bit 1 of IICA status register 0 (IICS0)
STT0: Bit 1 of IICA control register 00 (IICCTL00)

**Figure 13-21. Arbitration Timing Example**

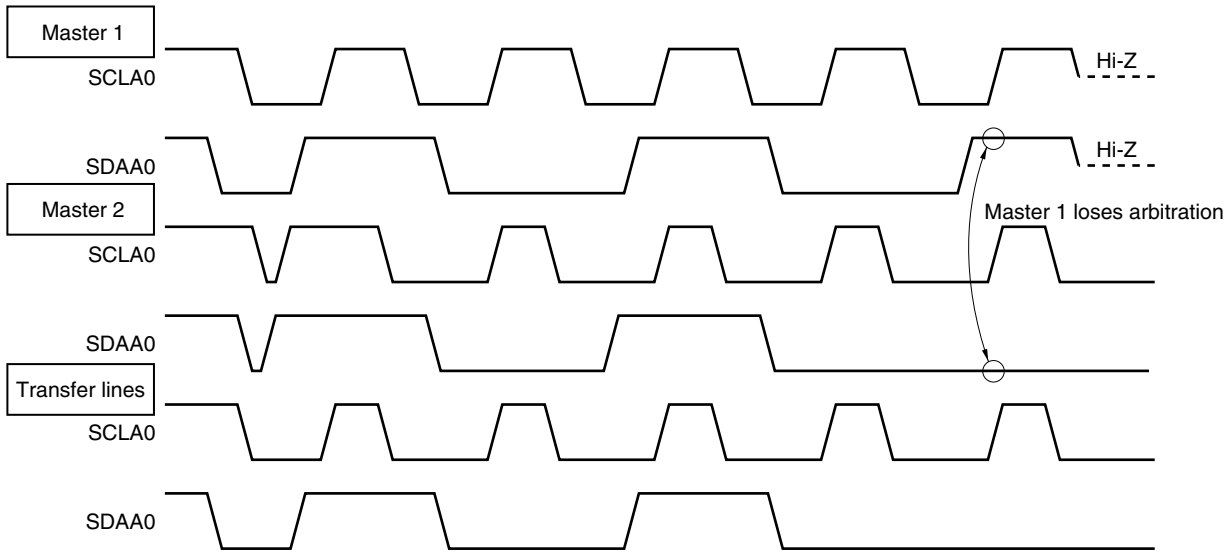


**Table 13-4. Status During Arbitration and Interrupt Request Generation Timing**

| Status During Arbitration | Interrupt Request Generation Timing |
|---|---|
| During address transmission | At falling edge of eighth or ninth clock following byte transfer[Note 1] |
| Read/write data after address transmission | |
| During extension code transmission | |
| Read/write data after extension code transmission | |
| During data transmission | |
| During $\overline{\text{ACK}}$ transfer period after data transmission | |
| When restart condition is detected during data transfer | |
| When stop condition is detected during data transfer | When stop condition is generated (when SPIE0 = 1)[Note 2] |
| When data is at low level while attempting to generate a restart condition | At falling edge of eighth or ninth clock following byte transfer[Note 1] |
| When stop condition is detected while attempting to generate a restart condition | When stop condition is generated (when SPIE0 = 1)[Note 2] |
| When data is at low level while attempting to generate a stop condition | At falling edge of eighth or ninth clock following byte transfer[Note 1] |
| When SCLA0 is at low level while attempting to generate a restart condition | |

**Notes 1.** When the WTIM0 bit (bit 3 of IICA control register 00 (IICCTL00)) = 1, an interrupt request occurs at the falling edge of the ninth clock. When WTIM0 = 0 and the extension code's slave address is received, an interrupt request occurs at the falling edge of the eighth clock.

**2.** When there is a chance that arbitration will occur, set SPIE0 = 1 for master device operation.

**Remark** SPIE0: Bit 4 of IICA control register 00 (IICCTL00)

### <R> 13.5.13 Wakeup function

The $I^2C$ bus slave function is a function that generates an interrupt request signal (INTIICA0) when a local address and extension code have been received.

This function makes processing more efficient by preventing unnecessary INTIICA0 signal from occurring when addresses do not match.

When a start condition is detected, wakeup standby mode is set. This wakeup standby mode is in effect while addresses are transmitted due to the possibility that an arbitration loss may change the master device (which has generated a start condition) to a slave device.

To use the wakeup function in the STOP mode, set the WUP0 bit to 1. Addresses can be received regardless of the operation clock. An interrupt request signal (INTIICA0) is also generated when a local address and extension code have been received. Operation returns to normal operation by using an instruction to clear (0) the WUP0 bit after this interrupt has been generated.

Figure 13-22 shows the flow for setting WUP0 = 1 and Figure 13-23 shows the flow for setting WUP0 = 0 upon an address match.

**Figure 13-22. Flow When Setting WUP0 = 1**

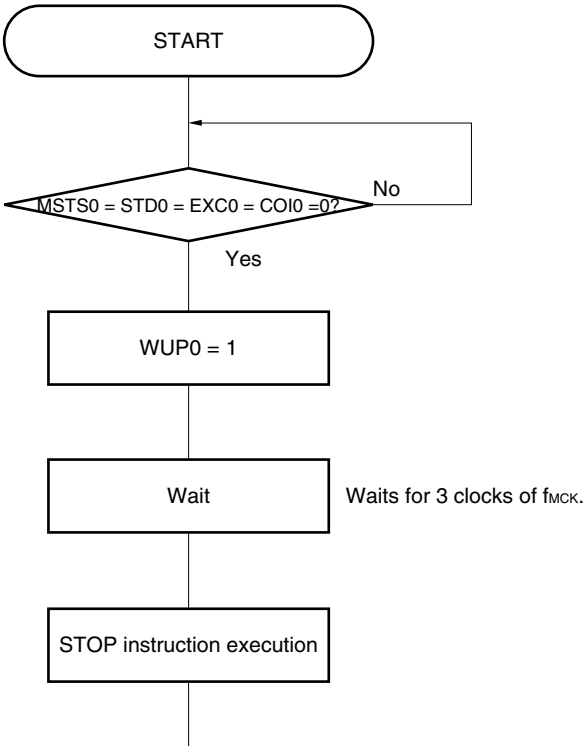
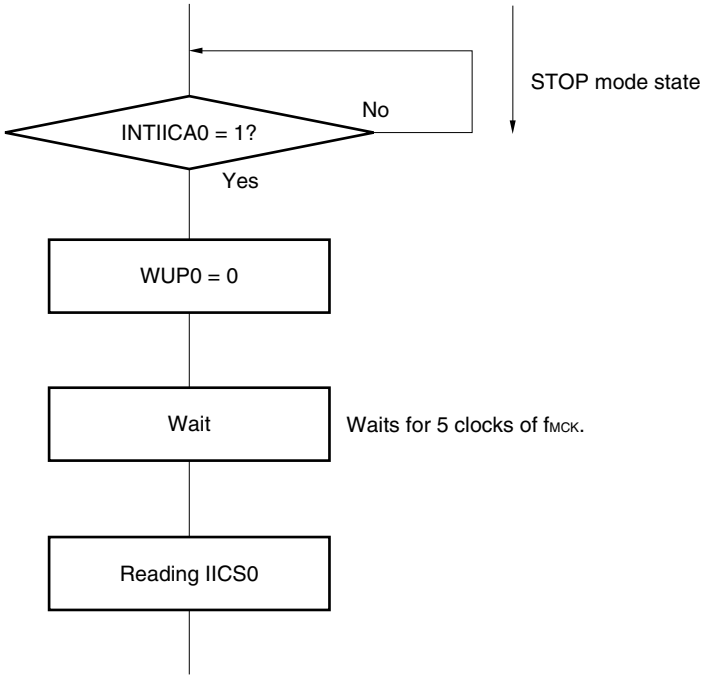


**Figure 13-23. Flow When Setting WUP0 = 0 Upon Address Match (Including Extension Code Reception)**



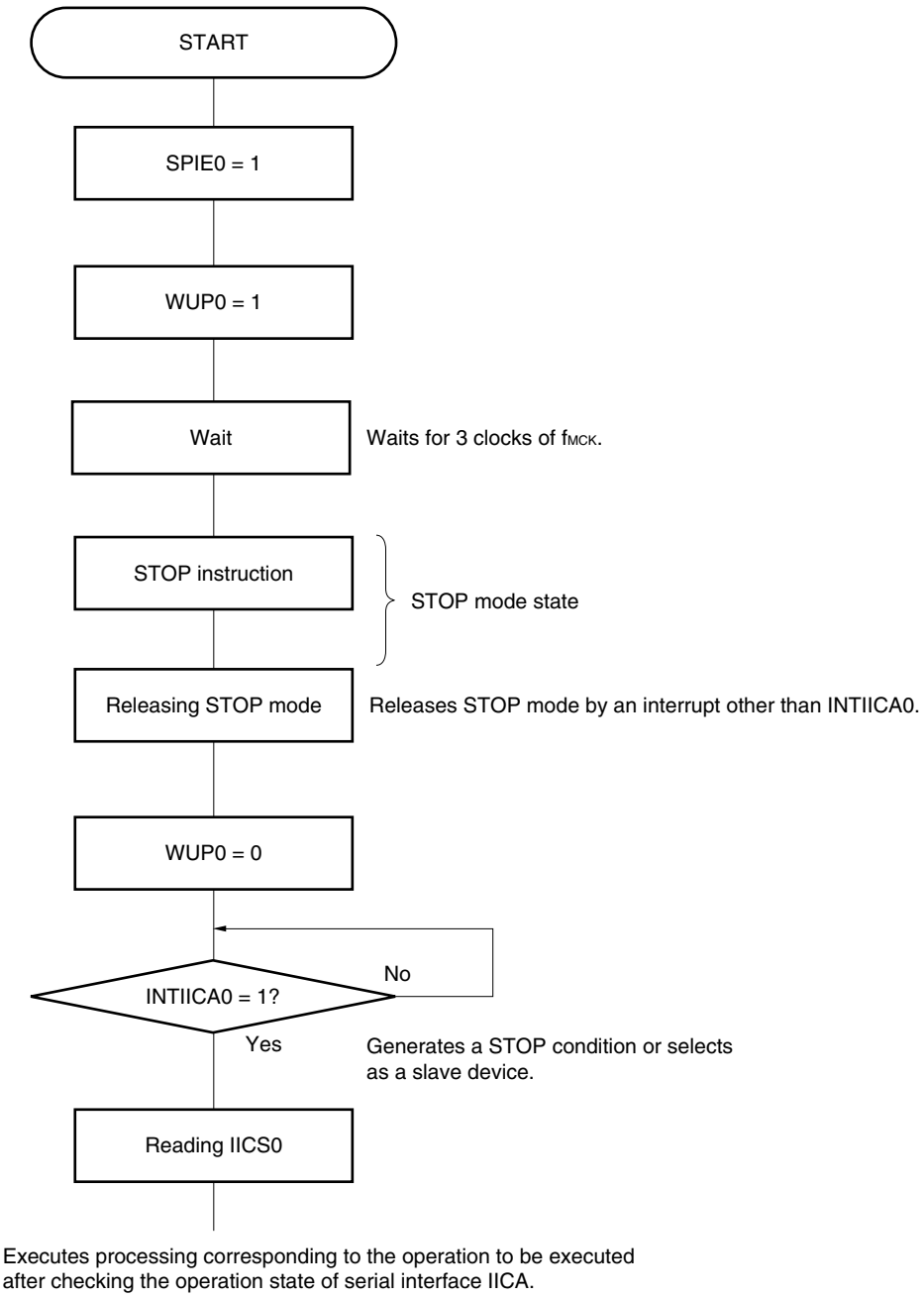

Executes processing corresponding to the operation to be executed after checking the operation state of serial interface IICA.

Use the following flows to perform the processing to release the STOP mode other than by an interrupt request (INTIICA0) generated from serial interface IICA.

- Master device operation: Flow shown in Figure 13-24
- Slave device operation: Same as the flow in Figure 13-23

**Figure 13-24. When Operating as Master Device After Releasing STOP Mode Other than by INTIICA0**

<R>



### 13.5.14 Communication reservation

**(1) When communication reservation function is enabled (bit 0 (IICRSV) of IICA flag register 0 (IICF0) = 0)**

To start master device communications when not currently using a bus, a communication reservation can be made to enable transmission of a start condition when the bus is released. There are two modes under which the bus is not used.

- When arbitration results in neither master nor slave operation
- When an extension code is received and slave operation is disabled ($\overline{ACK}$ is not returned and the bus was released by setting bit 6 (LREL0) of IICA control register 00 (IICCTL00) to 1 and saving communication).

If bit 1 (STT0) of the IICCTL00 register is set to 1 while the bus is not used (after a stop condition is detected), a start condition is automatically generated and wait state is set.
If an address is written to the IICA shift register 0 (IICA0) after bit 4 (SPIE0) of the IICCTL00 register was set to 1, and it was detected by generation of an interrupt request signal (INTIICA0) that the bus was released (detection of the stop condition), then the device automatically starts communication as the master. Data written to the IICA0 register before the stop condition is detected is invalid.
When the STT0 bit has been set to 1, the operation mode (as start condition or as communication reservation) is determined according to the bus status.

- If the bus has been released ..... a start condition is generated
- If the bus has not been released (standby mode)..... communication reservation

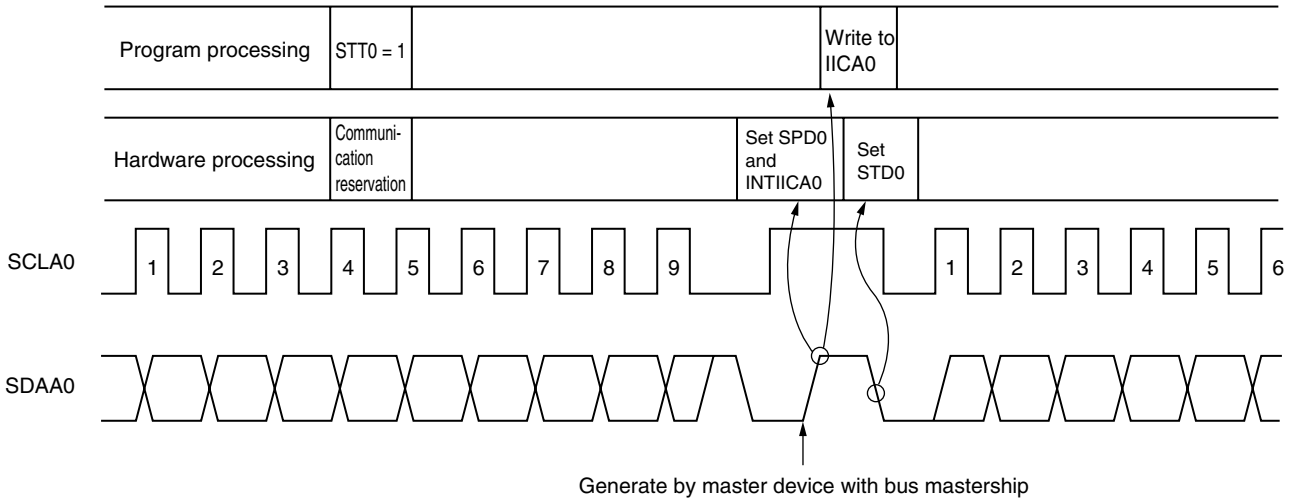
Check whether the communication reservation operates or not by using the MSTS0 bit (bit 7 of the IICA status register 0 (IICS0)) after the STT0 bit is set to 1 and the wait time elapses.
Use software to secure the wait time calculated by the following expression.

| Wait time from setting STT0 = 1 to checking the MSTS0 flag (number of clocks of $f_{MCK}$): |
|---|
| (IICWL0 setting value + IICWH0 setting value + 4) + $t_F \times 2 \times f_{MCK}$ [clocks] |

**Remark** IICWL0: IICA low-level width setting register 0
IICWH0: IICA high-level width setting register 0
$t_F$: SDAA0 and SCLA0 signal falling times
$f_{MCK}$: IICA operation clock frequency

Figure 13-25 shows the communication reservation timing.

**Figure 13-25. Communication Reservation Timing**




**Remark** IICA0: IICA shift register 0
STT0: Bit 1 of IICA control register 00 (IICCTL00)
STD0: Bit 1 of IICA status register 0 (IICS0)
SPD0: Bit 0 of IICA status register 0 (IICS0)

Communication reservations are accepted via the timing shown in Figure 13-26. After bit 1 (STD0) of the IICA status register 0 (IICS0) is set to 1, a communication reservation can be made by setting bit 1 (STT0) of IICA control register 00 (IICCTL00) to 1 before a stop condition is detected.

**Figure 13-26. Timing for Accepting Communication Reservations**

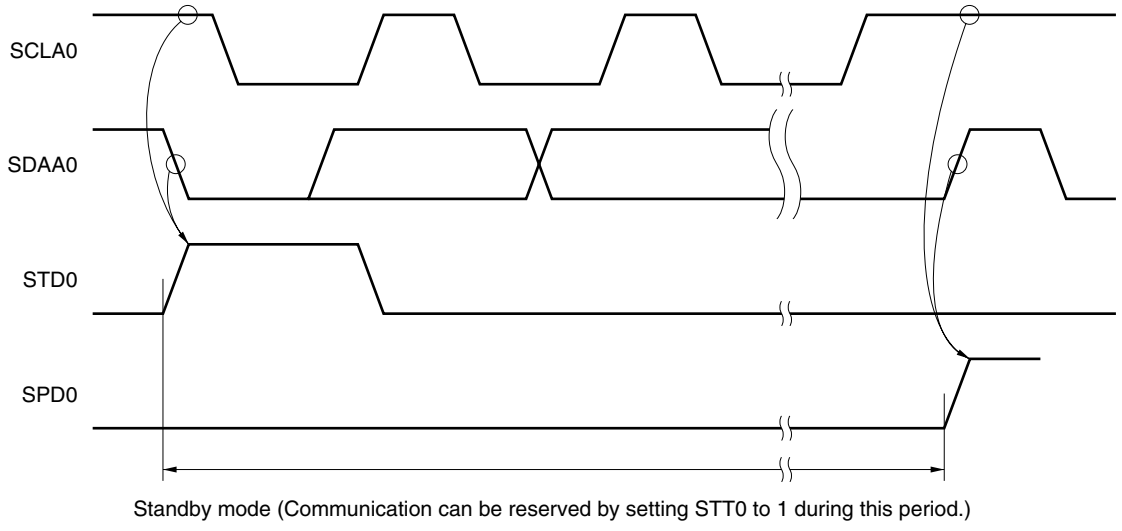
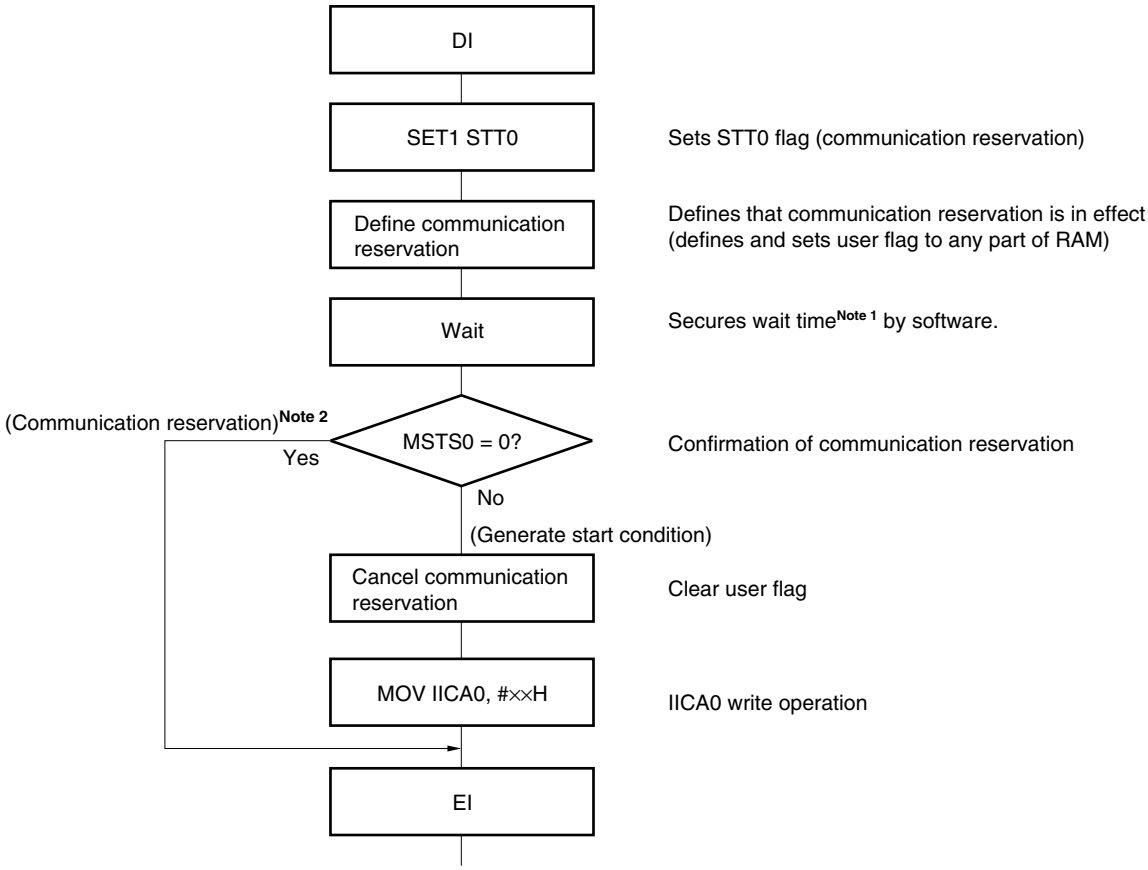



Figure 13-27 shows the communication reservation protocol.

**Figure 13-27. Communication Reservation Protocol**




**Notes 1.** The wait time ($f_{MCK}$) is calculated as follows.
(IICWL0 setting value + IICWH0 setting value + 4) + $t_F \times 2 \times f_{MCK}$ [clocks]

**2.** The communication reservation operation executes a write to the IICA shift register 0 (IICA0) when a stop condition interrupt request occurs.

**Remark** STT0: Bit 1 of IICA control register 00 (IICCTL00)
MSTS0: Bit 7 of IICA status register 0 (IICS0)
IICA0: IICA shift register 0
IICWL0: IICA low-level width setting register 0
IICWH0: IICA high-level width setting register 0
$t_F$: SDAA0 and SCLA0 signal falling times
$f_{MCK}$: IICA operation clock frequency

**(2) When communication reservation function is disabled (bit 0 (IICRSV) of IICA flag register 0 (IICF0) = 1)**

When bit 1 (STT0) of IICA control register 00 (IICCTL00) is set to 1 when the bus is not used in a communication during bus communication, this request is rejected and a start condition is not generated. The following two statuses are included in the status where bus is not used.

- When arbitration results in neither master nor slave operation
- When an extension code is received and slave operation is disabled ($\overline{ACK}$ is not returned and the bus was released by setting bit 6 (LREL0) of the IICCTL00 register to 1 and saving communication)

To confirm whether the start condition was generated or request was rejected, check STCF (bit 7 of the IICF0 register). It takes up to 5 clocks of $f_{MCK}$ until the STCF bit is set to 1 after setting STT0 = 1. Therefore, secure the time by software.

### 13.5.15 Cautions

(1) When STCEN = 0

Immediately after $I^2C$ operation is enabled (IICE0 = 1), the bus communication status (IICBSY = 1) is recognized regardless of the actual bus status. When changing from a mode in which no stop condition has been detected to a master device communication mode, first generate a stop condition to release the bus, then perform master device communication.

When using multiple masters, it is not possible to perform master device communication when the bus has not been released (when a stop condition has not been detected).

Use the following sequence for generating a stop condition.

<1> Set IICA control register 01 (IICCTL01).
<2> Set bit 7 (IICE0) of IICA control register 00 (IICCTL00) to 1.
<3> Set bit 0 (SPT0) of the IICCTL00 register to 1.

(2) When STCEN = 1

Immediately after $I^2C$ operation is enabled (IICE0 = 1), the bus released status (IICBSY = 0) is recognized regardless of the actual bus status. To generate the first start condition (STT0 = 1), it is necessary to confirm that the bus has been released, so as to not disturb other communications.

(3) If other $I^2C$ communications are already in progress

If $I^2C$ operation is enabled and the device participates in communication already in progress when the SDAA0 pin is low and the SCLA0 pin is high, the macro of $I^2C$ recognizes that the SDAA0 pin has gone low (detects a start condition). If the value on the bus at this time can be recognized as an extension code, $\overline{ACK}$ is returned, but this interferes with other $I^2C$ communications. To avoid this, start $I^2C$ in the following sequence.

<1> Clear bit 4 (SPIE0) of the IICCTL00 register to 0 to disable generation of an interrupt request signal (INTIICA0) when the stop condition is detected.
<2> Set bit 7 (IICE0) of the IICCTL00 register to 1 to enable the operation of $I^2C$.
<3> Wait for detection of the start condition.
<R> <4> Set bit 6 (LREL0) of the IICCTL00 register to 1 before $\overline{ACK}$ is returned (4 to 72 clocks of $f_{MCK}$ after setting the IICE0 bit to 1), to forcibly disable detection.

(4) Setting the STT0 and SPT0 bits (bits 1 and 0 of the IICCTL00 register) again after they are set and before they are cleared to 0 is prohibited.

(5) When transmission is reserved, set the SPIE0 bit (bit 4 of the IICTL0 register) to 1 so that an interrupt request is generated when the stop condition is detected. Transfer is started when communication data is written to the IICA shift register 0 (IICA0) after the interrupt request is generated. Unless the interrupt is generated when the stop condition is detected, the device stops in the wait state because the interrupt request is not generated when communication is started. However, it is not necessary to set the SPIE0 bit to 1 when the MSTS0 bit (bit 7 of the IICA status register 0 (IICS0)) is detected by software.

### 13.5.16 Communication operations

The following shows three operation procedures with the flowchart.

#### (1) Master operation in single master system

The flowchart when using the RL78/G1A as the master in a single master system is shown below.

This flowchart is broadly divided into the initial settings and communication processing. Execute the initial settings at startup. If communication with the slave is required, prepare the communication and then execute communication processing.

#### (2) Master operation in multimaster system

In the $I^2C$ bus multimaster system, whether the bus is released or used cannot be judged by the $I^2C$ bus specifications when the bus takes part in a communication. Here, when data and clock are at a high level for a certain period (1 frame), the RL78/G1A takes part in a communication with bus released state.

This flowchart is broadly divided into the initial settings, communication waiting, and communication processing. The processing when the RL78/G1A looses in arbitration and is specified as the slave is omitted here, and only the processing as the master is shown. Execute the initial settings at startup to take part in a communication. Then, wait for the communication request as the master or wait for the specification as the slave. The actual communication is performed in the communication processing, and it supports the transmission/reception with the slave and the arbitration with other masters.

#### (3) Slave operation

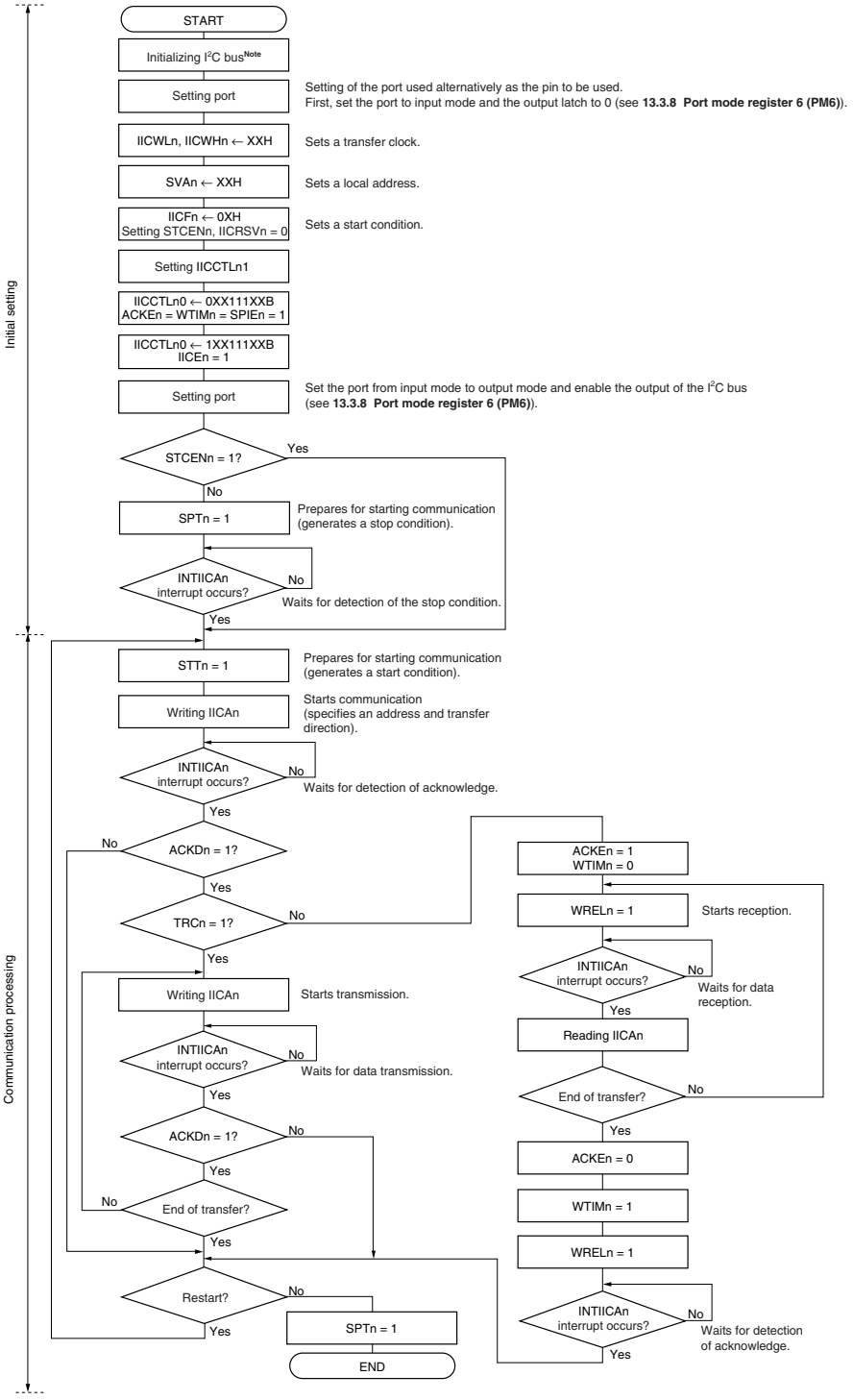
An example of when the RL78/G1A is used as the $I^2C$ bus slave is shown below.

When used as the slave, operation is started by an interrupt. Execute the initial settings at startup, then wait for the INTIICA0 interrupt occurrence (communication waiting). When an INTIICA0 interrupt occurs, the communication status is judged and its result is passed as a flag over to the main processing.

By checking the flags, necessary communication processing is performed.

### (1) Master operation in single-master system

**Figure 13-28. Master Operation in Single-Master System**



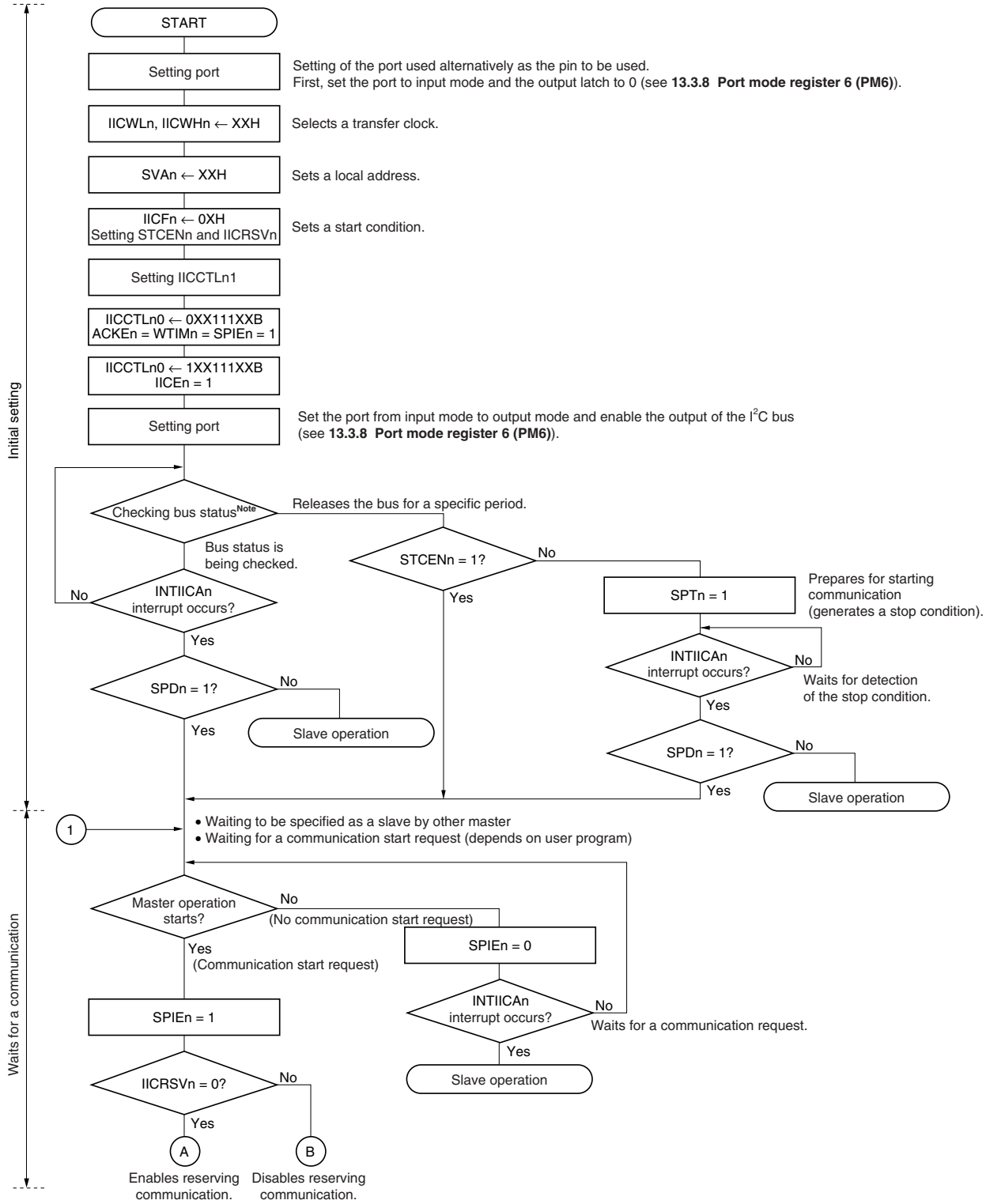

**Note** Release (SCLA0 and SDAA0 pins = high level) the I²C bus in conformance with the specifications of the product that is communicating. If EEPROM is outputting a low level to the SDAA0 pin, for example, set the SCLA0 pin in the output port mode, and output a clock pulse from the output port until the SDAA0 pin is constantly at high level.

**Remarks 1.** Conform to the specifications of the product that is communicating, with respect to the transmission and reception formats.

**2.** n = 0

### (2) Master operation in multi-master system

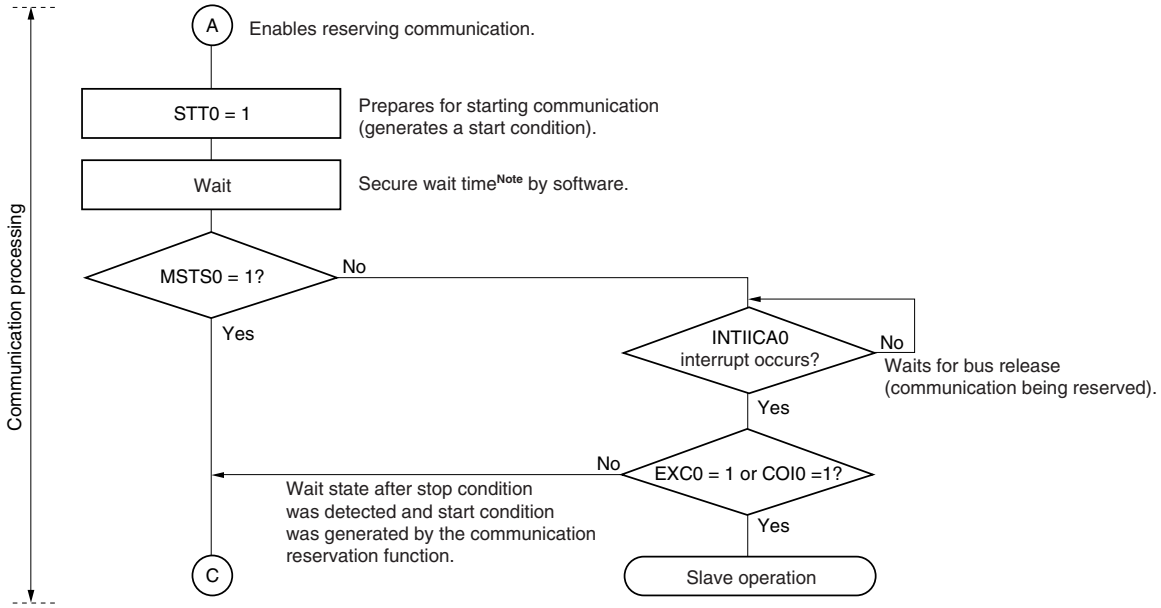
**Figure 13-29. Master Operation in Multi-Master System (1/4)**




**Note** Confirm that the bus is released (CLD0 bit = 1, DAD0 bit = 1) for a specific period (for example, for a period of one frame). If the SDAA0 pin is constantly at low level, decide whether to release the $I^2C$ bus (SCLA0 and SDAA0 pins = high level) in conformance with the specifications of the product that is communicating.

**Remark** n = 0

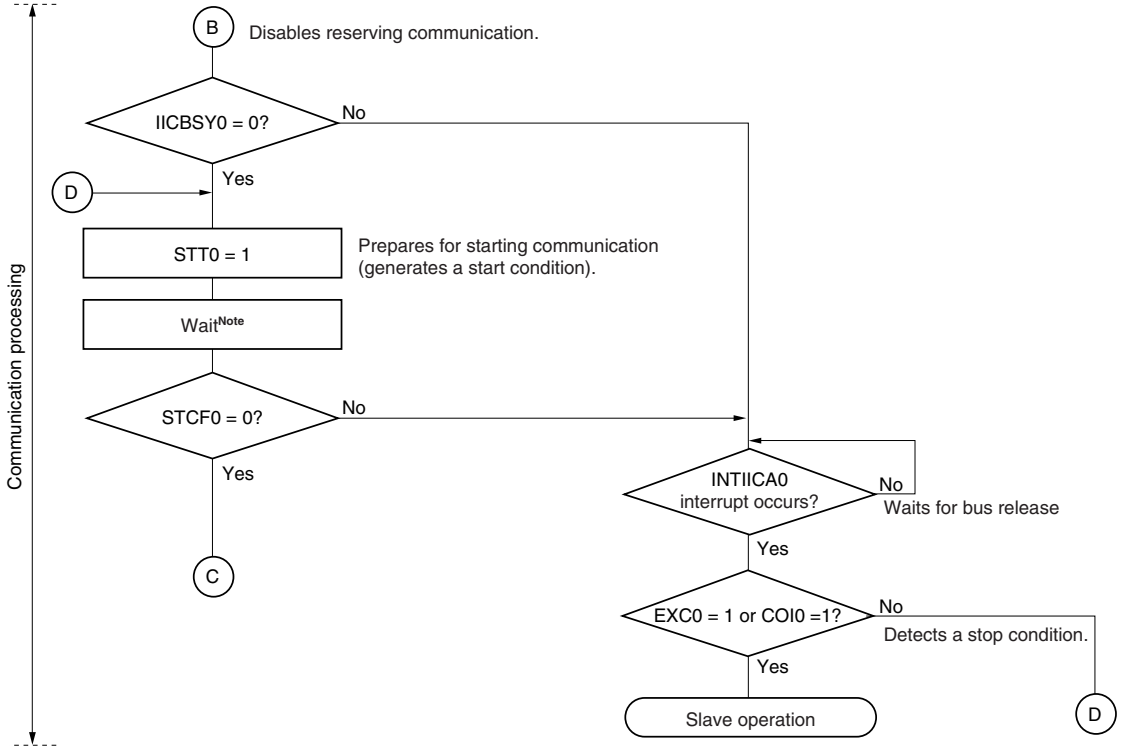
**Figure 13-29. Master Operation in Multi-Master System (2/4)**




<R> **Note** The wait time (number of clocks of $f_{MCK}$) is calculated as follows.
(IICWL0 setting value + IICWH0 setting value + 4) + $t_F \times 2 \times f_{MCK}$ [clocks]

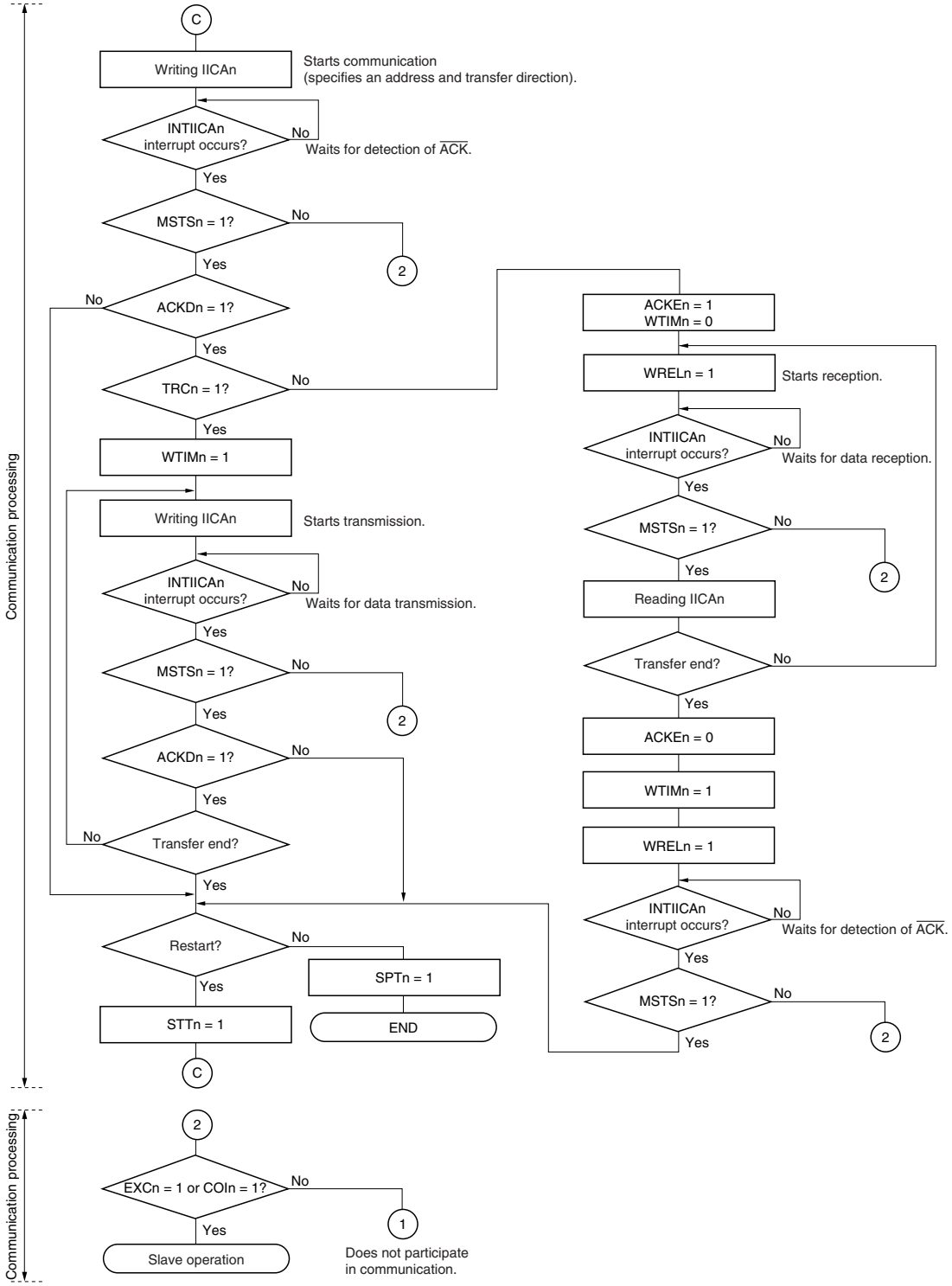
**Remark** IICWL0: IICA low-level width setting register 0
IICWH0: IICA high-level width setting register 0
$t_F$: SDAA0 and SCLA0 signal falling times
$f_{MCK}$: IICA operation clock frequency

**Figure 13-29. Master Operation in Multi-Master System (3/4)**




<R> **Note** Wait: Wait for five clocks of $f_{MCL}$

**Figure 13-29. Master Operation in Multi-Master System (4/4)**




<R>
<R>
<R>

**Remarks 1.** Conform to the specifications of the product that is communicating, with respect to the transmission and reception formats.

**2.** To use the device as a master in a multi-master system, read the MSTS0 bit each time interrupt INTIICA0 has occurred to check the arbitration result.

**3.** To use the device as a slave in a multi-master system, check the status by using the IICA status register 0 (IICS0) and IICA flag register 0 (IICF0) each time interrupt INTIICA0 has occurred, and determine the processing to be performed next.

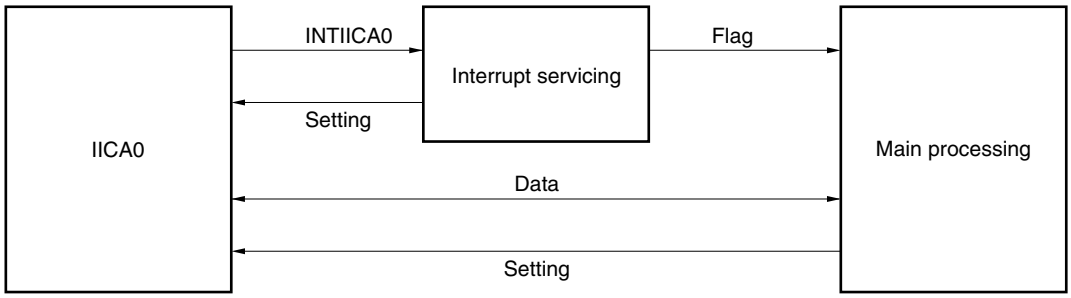
**4.** n = 0

### (3) Slave operation

The processing procedure of the slave operation is as follows.

Basically, the slave operation is event-driven. Therefore, processing by the INTIICA0 interrupt (processing that must substantially change the operation status such as detection of a stop condition during communication) is necessary.

In the following explanation, it is assumed that the extension code is not supported for data communication. It is also assumed that the INTIICA0 interrupt servicing only performs status transition processing, and that actual data communication is performed by the main processing.




Therefore, data communication processing is performed by preparing the following three flags and passing them to the main processing instead of INTIICA0.

**<1> Communication mode flag**

This flag indicates the following two communication statuses.

- Clear mode: Status in which data communication is not performed
- Communication mode: Status in which data communication is performed (from valid address detection to stop condition detection, no detection of $\overline{\text{ACK}}$ from master, address mismatch)

**<2> Ready flag**

This flag indicates that data communication is enabled. Its function is the same as the INTIICA0 interrupt for ordinary data communication. This flag is set by interrupt servicing and cleared by the main processing. Clear this flag by interrupt servicing when communication is started. However, the ready flag is not set by interrupt servicing when the first data is transmitted. Therefore, the first data is transmitted without the flag being cleared (an address match is interpreted as a request for the next data).

**<3> Communication direction flag**

This flag indicates the direction of communication. Its value is the same as the TRC0 bit.

The main processing of the slave operation is explained next.

Start serial interface IICA and wait until communication is enabled. When communication is enabled, execute communication by using the communication mode flag and ready flag (processing of the stop condition and start condition is performed by an interrupt. Here, check the status by using the flags).

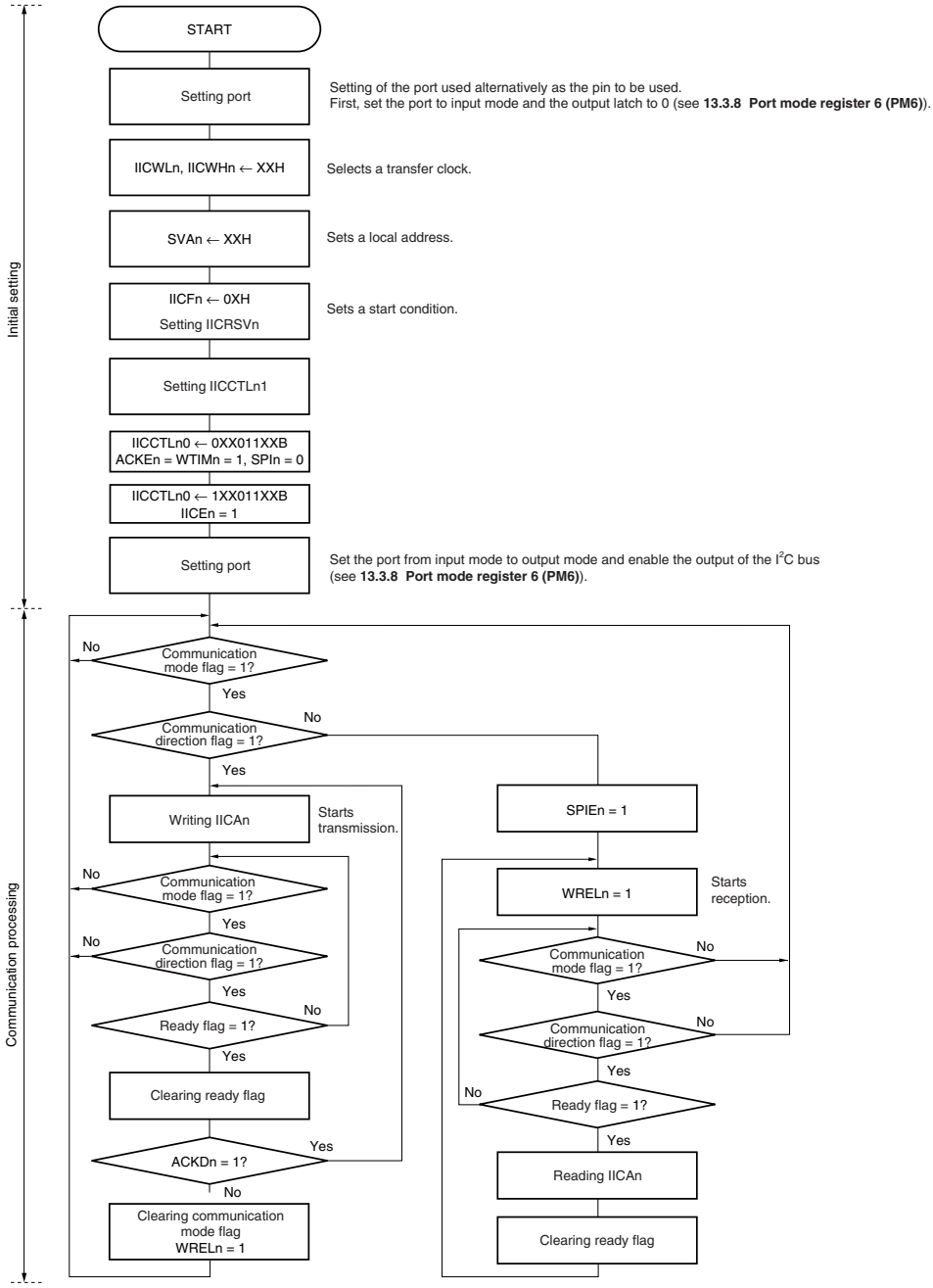
The transmission operation is repeated until the master no longer returns $\overline{ACK}$. If $\overline{ACK}$ is not returned from the master, communication is completed.

For reception, the necessary amount of data is received. When communication is completed, $\overline{ACK}$ is not returned as the next data. After that, the master generates a stop condition or restart condition. Exit from the communication status occurs in this way.

**Figure 13-30. Slave Operation Flowchart (1)**

<R>

<R>




**Remarks 1.** Conform to the specifications of the product that is in communication, regarding the transmission and reception formats.

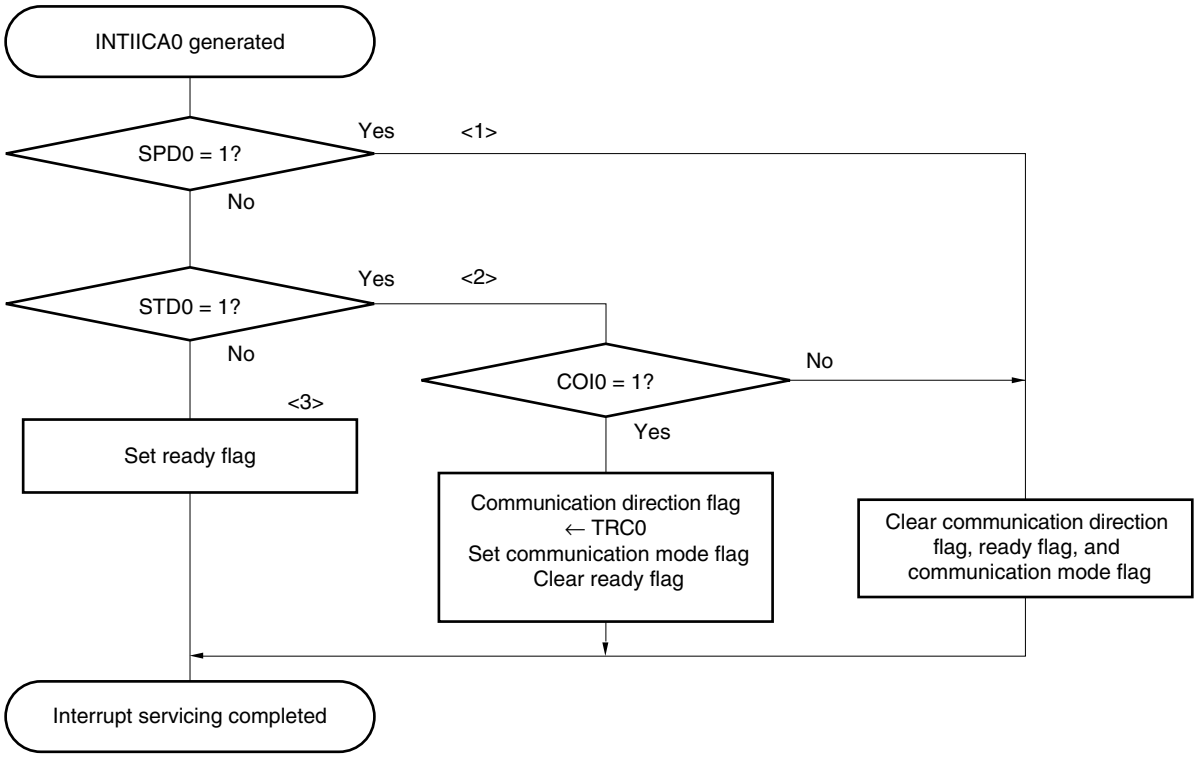
**2.** n = 0

An example of the processing procedure of the slave with the INTIICA0 interrupt is explained below (processing is performed assuming that no extension code is used). The INTIICA0 interrupt checks the status, and the following operations are performed.

<1> Communication is stopped if the stop condition is issued.
<2> If the start condition is issued, the address is checked and communication is completed if the address does not match. If the address matches, the communication mode is set, wait is cancelled, and processing returns from the interrupt (the ready flag is cleared).
<3> For data transmit/receive, only the ready flag is set. Processing returns from the interrupt with the $I^2C$ bus remaining in the wait state.

**Remark** <1> to <3> above correspond to <1> to <3> in Figure 13-31 Slave Operation Flowchart (2).

**Figure 13-31. Slave Operation Flowchart (2)**




### 13.5.17 Timing of $I^2C$ interrupt request (INTIICA0) occurrence

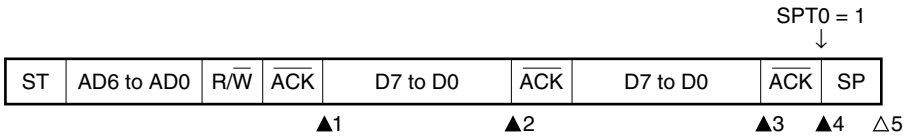
The timing of transmitting or receiving data and generation of interrupt request signal INTIICA0, and the value of the IICA status register 0 (IICS0) when the INTIICA0 signal is generated are shown below.

**Remark** ST: Start condition
AD6 to AD0: Address
R/$\overline{W}$: Transfer direction specification
$\overline{ACK}$: Acknowledge
D7 to D0: Data
SP: Stop condition

### (1) Master device operation

#### (a) Start ~ Address ~ Data ~ Data ~ Stop (transmission/reception)

##### (i) When WTIM0 = 0

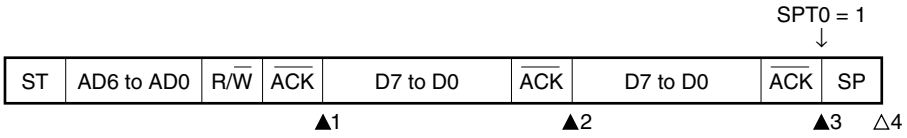



▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000×000B
▲3: IICS0 = 1000×000B (Sets the WTIM0 bit to 1)[Note]
▲4: IICS0 = 1000××00B (Sets the SPT0 bit to 1)[Note]
△5: IICS0 = 00000001B

**Note** To generate a stop condition, set the WTIM0 bit to 1 and change the timing for generating the INTIICA0 interrupt request signal.

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

##### (ii) When WTIM0 = 1

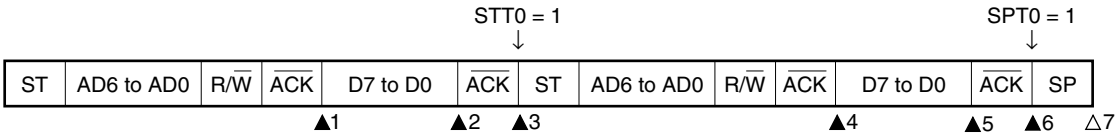



▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000×100B
▲3: IICS0 = 1000××00B (Sets the SPT0 bit to 1)
△4: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(b) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop (restart)**

**(i) When WTIM0 = 0**

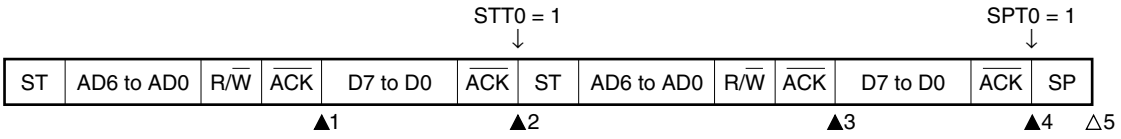



▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000×000B (Sets the WTIM0 bit to 1)Note 1
▲3: IICS0 = 1000××00B (Clears the WTIM0 bit to 0Note 2, sets the STT0 bit to 1)
▲4: IICS0 = 1000×110B
▲5: IICS0 = 1000×000B (Sets the WTIM0 bit to 1)Note 3
▲6: IICS0 = 1000××00B (Sets the SPT0 bit to 1)
△7: IICS0 = 00000001B

**Notes 1.** To generate a start condition, set the WTIM0 bit to 1 and change the timing for generating the INTIICA0 interrupt request signal.
**2.** Clear the WTIM0 bit to 0 to restore the original setting.
**3.** To generate a stop condition, set the WTIM0 bit to 1 and change the timing for generating the INTIICA0 interrupt request signal.

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(ii) When WTIM0 = 1**

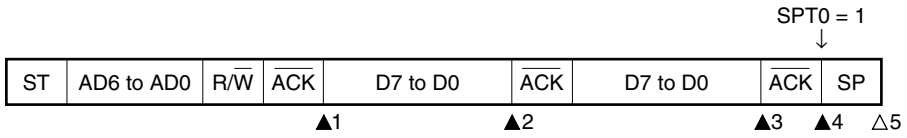



▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000××00B (Sets the STT0 bit to 1)
▲3: IICS0 = 1000×110B
▲4: IICS0 = 1000××00B (Sets the SPT0 bit to 1)
△5: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(c) Start ~ Code ~ Data ~ Data ~ Stop (extension code transmission)**

**(i) When WTIM0 = 0**

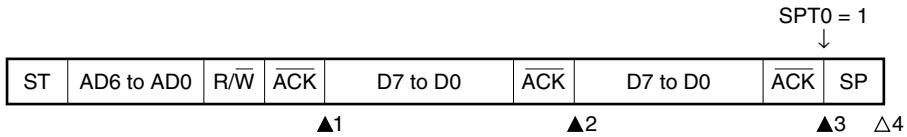



▲1: IICS0 = 1010×110B
▲2: IICS0 = 1010×000B
▲3: IICS0 = 1010×000B (Sets the WTIM0 bit to 1)[Note]
▲4: IICS0 = 1010××00B (Sets the SPT0 bit to 1)
△5: IICS0 = 00000001B

**Note** To generate a stop condition, set the WTIM0 bit to 1 and change the timing for generating the INTIICA0 interrupt request signal.

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(ii) When WTIM0 = 1**



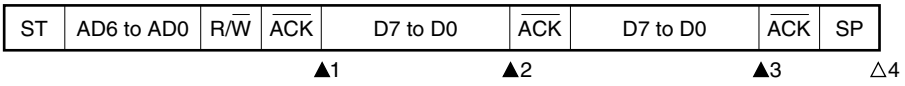

▲1: IICS0 = 1010×110B
▲2: IICS0 = 1010×100B
▲3: IICS0 = 1010××00B (Sets the SPT0 bit to 1)
△4: IICS0 = 00001001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(2) Slave device operation (slave address data reception)**

**(a) Start ~ Address ~ Data ~ Data ~ Stop**

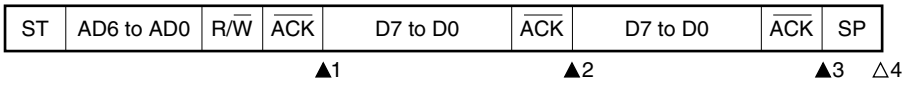
**(i) When WTIM0 = 0**




▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001×000B
▲3: IICS0 = 0001×000B
△4: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(ii) When WTIM0 = 1**

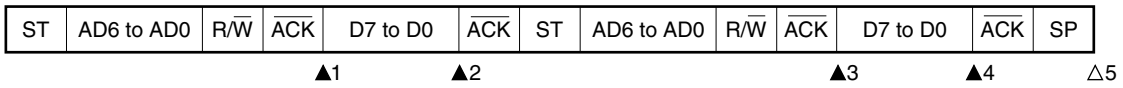



▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001×100B
▲3: IICS0 = 0001××00B
△4: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(b) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop**

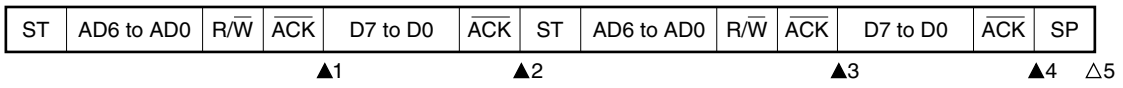
**(i) When WTIM0 = 0 (after restart, matches with SVA0)**




▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001×000B
▲3: IICS0 = 0001×110B
▲4: IICS0 = 0001×000B
△5: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(ii) When WTIM0 = 1 (after restart, matches with SVA0)**

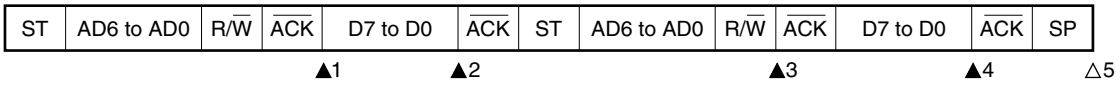



▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001××00B
▲3: IICS0 = 0001×110B
▲4: IICS0 = 0001××00B
△5: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(c) Start ~ Address ~ Data ~ Start ~ Code ~ Data ~ Stop**

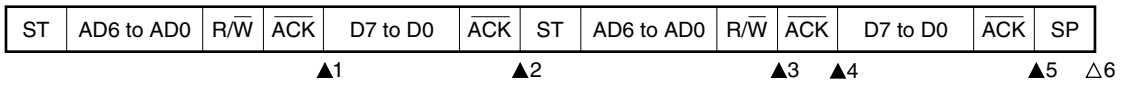
**(i) When WTIM0 = 0 (after restart, does not match address (= extension code))**




▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001×000B
▲3: IICS0 = 0010×010B
▲4: IICS0 = 0010×000B
△5: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(ii) When WTIM0 = 1 (after restart, does not match address (= extension code))**

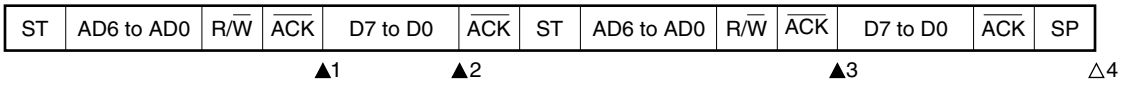



▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001××00B
▲3: IICS0 = 0010×010B
▲4: IICS0 = 0010×110B
▲5: IICS0 = 0010××00B
△6: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(d) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop**

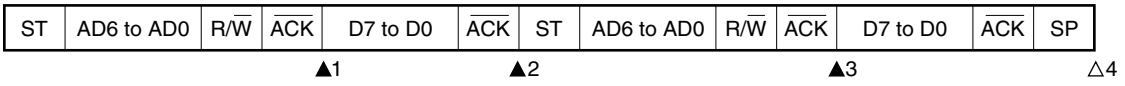
**(i) When WTIM0 = 0 (after restart, does not match address (= not extension code))**




▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001×000B
▲3: IICSn = 00000×10B
△4: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(ii) When WTIM0 = 1 (after restart, does not match address (= not extension code))**




▲1: IICS0 = 0001×110B
▲2: IICS0 = 0001××00B
▲3: IICSn = 00000×10B
△4: IICS0 = 00000001B

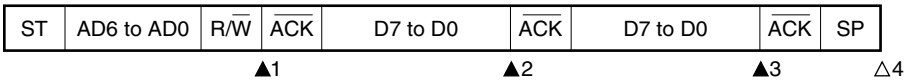
**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(3) Slave device operation (when receiving extension code)**

The device is always participating in communication when it receives an extension code.

**(a) Start ~ Code ~ Data ~ Data ~ Stop**

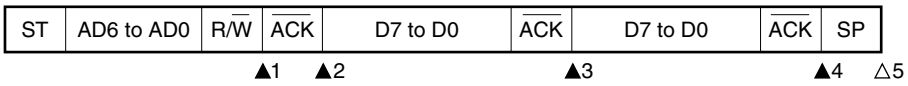
**(i) When WTIM0 = 0**




▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×000B
▲3: IICS0 = 0010×000B
△4: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(ii) When WTIM0 = 1**

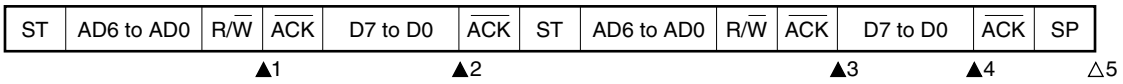



▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×110B
▲3: IICS0 = 0010×100B
▲4: IICS0 = 0010××00B
△5: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(b) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop**

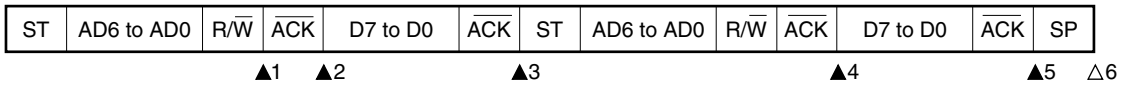
**(i) When WTIM0 = 0 (after restart, matches SVA0)**




▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×000B
▲3: IICS0 = 0001×110B
▲4: IICS0 = 0001×000B
△5: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(ii) When WTIM0 = 1 (after restart, matches SVA0)**

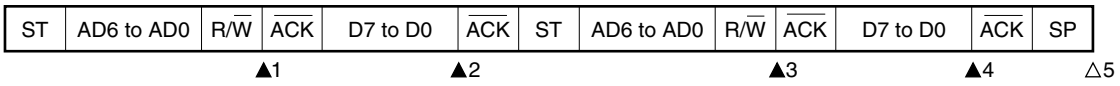



▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×110B
▲3: IICS0 = 0010××00B
▲4: IICS0 = 0001×110B
▲5: IICS0 = 0001××00B
△6: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(c) Start ~ Code ~ Data ~ Start ~ Code ~ Data ~ Stop**

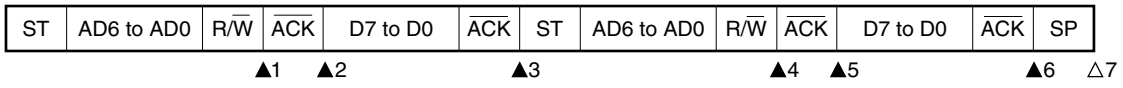
**(i) When WTIM0 = 0 (after restart, extension code reception)**




▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×000B
▲3: IICS0 = 0010×010B
▲4: IICS0 = 0010×000B
△5: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(ii) When WTIM0 = 1 (after restart, extension code reception)**

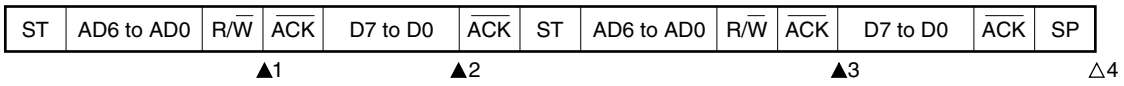



▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×110B
▲3: IICS0 = 0010××00B
▲4: IICS0 = 0010×010B
▲5: IICS0 = 0010×110B
▲6: IICS0 = 0010××00B
△7: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(d) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop**

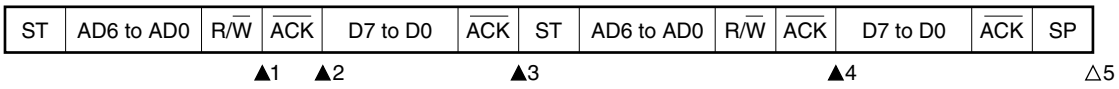
**(i) When WTIM0 = 0 (after restart, does not match address (= not extension code))**




▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×000B
▲3: IICS0 = 00000×10B
△4: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(ii) When WTIM0 = 1 (after restart, does not match address (= not extension code))**

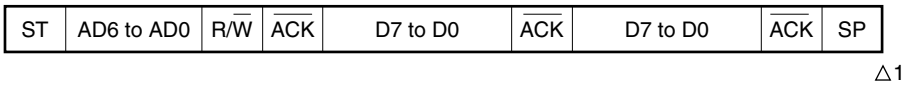



▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×110B
▲3: IICS0 = 0010××00B
▲4: IICS0 = 00000×10B
△5: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(4) Operation without communication**

**(a) Start ~ Code ~ Data ~ Data ~ Stop**




△1: IICS0 = 00000001B

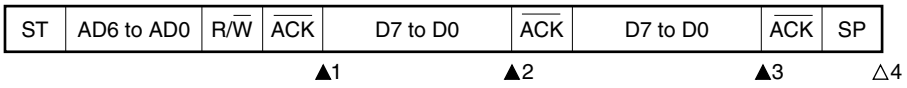
**Remark** △: Generated only when SPIE0 = 1

**(5) Arbitration loss operation (operation as slave after arbitration loss)**

When the device is used as a master in a multi-master system, read the MSTS0 bit each time interrupt request signal INTIICA0 has occurred to check the arbitration result.

**(a) When arbitration loss occurs during transmission of slave address data**

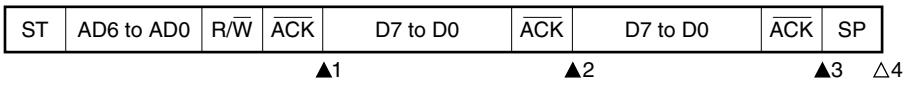
**(i) When WTIM0 = 0**




▲1: IICS0 = 0101×110B
▲2: IICS0 = 0001×000B
▲3: IICS0 = 0001×000B
△4: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(ii) When WTIM0 = 1**



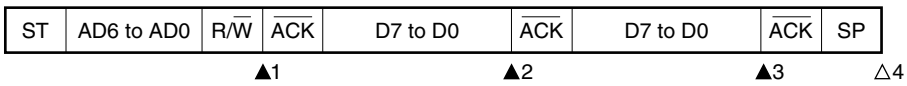

▲1 ▲2 ▲3 △4

▲1: IICS0 = 0101×110B
▲2: IICS0 = 0001×100B
▲3: IICS0 = 0001××00B
△4: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(b) When arbitration loss occurs during transmission of extension code**

**(i) When WTIM0 = 0**

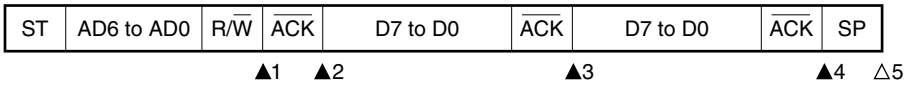



▲1 ▲2 ▲3 △4

▲1: IICS0 = 0110×010B
▲2: IICS0 = 0010×000B
▲3: IICS0 = 0010×000B
△4: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(ii) When WTIM0 = 1**



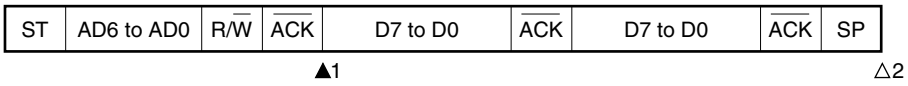

▲1: IICS0 = 0110×010B
▲2: IICS0 = 0010×110B
▲3: IICS0 = 0010×100B
▲4: IICS0 = 0010××00B
△5: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(6) Operation when arbitration loss occurs (no communication after arbitration loss)**

When the device is used as a master in a multi-master system, read the MSTS0 bit each time interrupt request signal INTIICA0 has occurred to check the arbitration result.

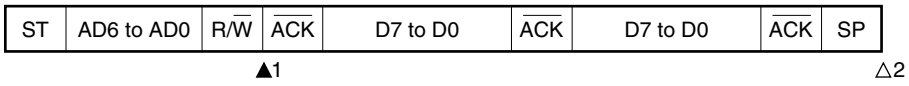
**(a) When arbitration loss occurs during transmission of slave address data (when WTIM0 = 1)**




▲1: IICS0 = 01000110B
△2: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1

**(b) When arbitration loss occurs during transmission of extension code**

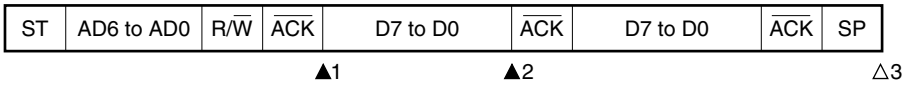



▲1: IICS0 = 0110×010B
Sets LREL0 = 1 by software
△2: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(c) When arbitration loss occurs during transmission of data**

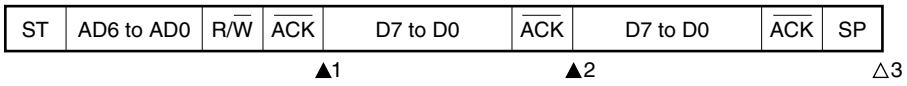
**(i) When WTIM0 = 0**




▲1: IICS0 = 10001110B
▲2: IICS0 = 01000000B
△3: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1

**(ii) When WTIM0 = 1**

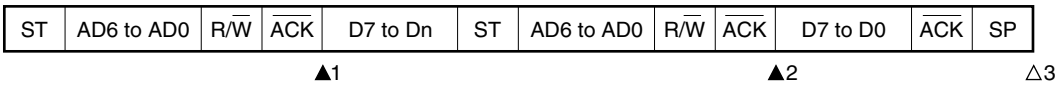



▲1: IICS0 = 10001110B
▲2: IICS0 = 01000100B
△3: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1

**(d) When loss occurs due to restart condition during data transfer**

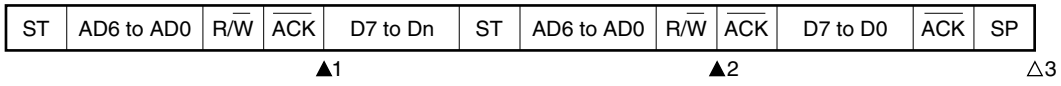
**(i) Not extension code (Example: unmatches with SVA0)**




▲1: IICS0 = 1000×110B
▲2: IICS0 = 01000110B
△3: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care
n = 6 to 0

**(ii) Extension code**




▲1: IICS0 = 1000×110B

▲2: IICS0 = 01100010B

Sets LREL0 = 1 by software

△3: IICS0 = 00000001B

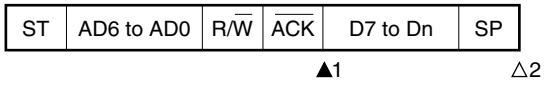
**Remark** ▲: Always generated

△: Generated only when SPIE0 = 1

×: Don't care

n = 6 to 0

**(e) When loss occurs due to stop condition during data transfer**




▲1: IICS0 = 10000110B

△2: IICS0 = 01000001B

**Remark** ▲: Always generated

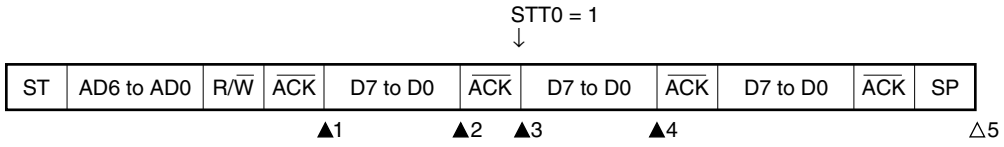
△: Generated only when SPIE0 = 1

×: Don't care

n = 6 to 0

**(f) When arbitration loss occurs due to low-level data when attempting to generate a restart condition**

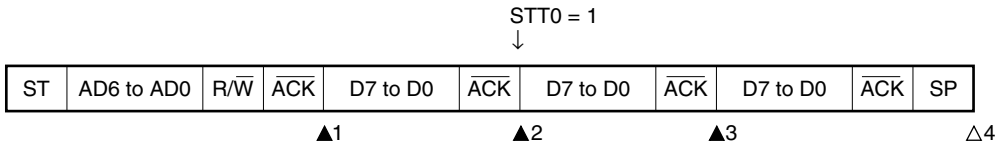
**(i) When WTIM0 = 0**




▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000×000B (Sets the WTIM0 bit to 1)
▲3: IICS0 = 1000×100B (Clears the WTIM0 bit to 0)
▲4: IICS0 = 01000000B
△5: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(ii) When WTIM0 = 1**

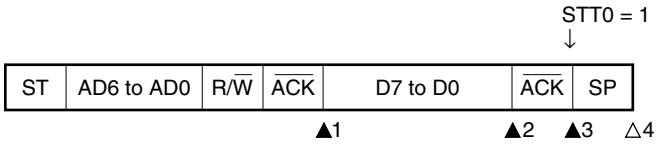



▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000×100B (Sets the STT0 bit to 1)
▲3: IICS0 = 01000100B
△4: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(g) When arbitration loss occurs due to a stop condition when attempting to generate a restart condition**

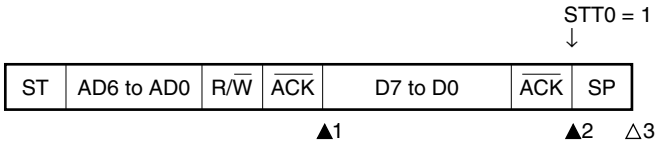
**(i) When WTIM0 = 0**




▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000×000B (Sets the WTIM0 bit to 1)
▲3: IICS0 = 1000××00B (Sets the STT0 bit to 1)
△4: IICS0 = 01000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(ii) When WTIM0 = 1**

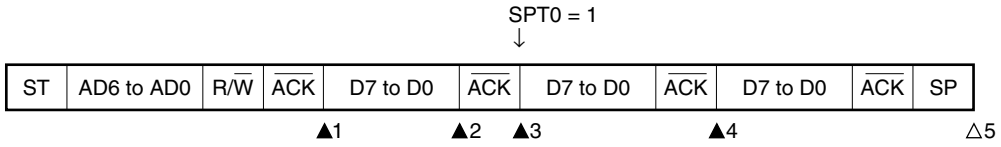



▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000××00B (Sets the STT0 bit to 1)
△3: IICS0 = 01000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(h) When arbitration loss occurs due to low-level data when attempting to generate a stop condition**

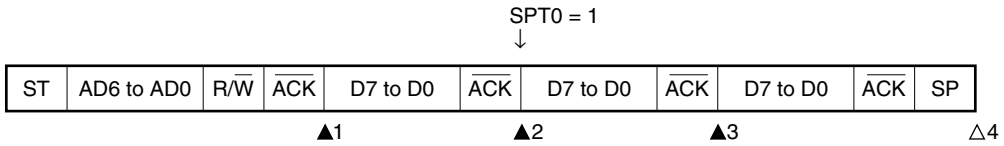
**(i) When WTIM0 = 0**




▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000×000B (Sets the WTIM0 bit to 1)
▲3: IICS0 = 1000×100B (Clears the WTIM0 bit to 0)
▲4: IICS0 = 01000100B
△5: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(ii) When WTIM0 = 1**




▲1: IICS0 = 1000×110B
▲2: IICS0 = 1000×100B (Sets the SPT0 bit to 1)
▲3: IICS0 = 01000100B
△4: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

## 13.6 Timing Charts

When using the $I^2C$ bus mode, the master device outputs an address via the serial bus to select one of several slave devices as its communication partner.

After outputting the slave address, the master device transmits the TRC0 bit (bit 3 of the IICA status register 0 (IICS0)), which specifies the data transfer direction, and then starts serial communication with the slave device.

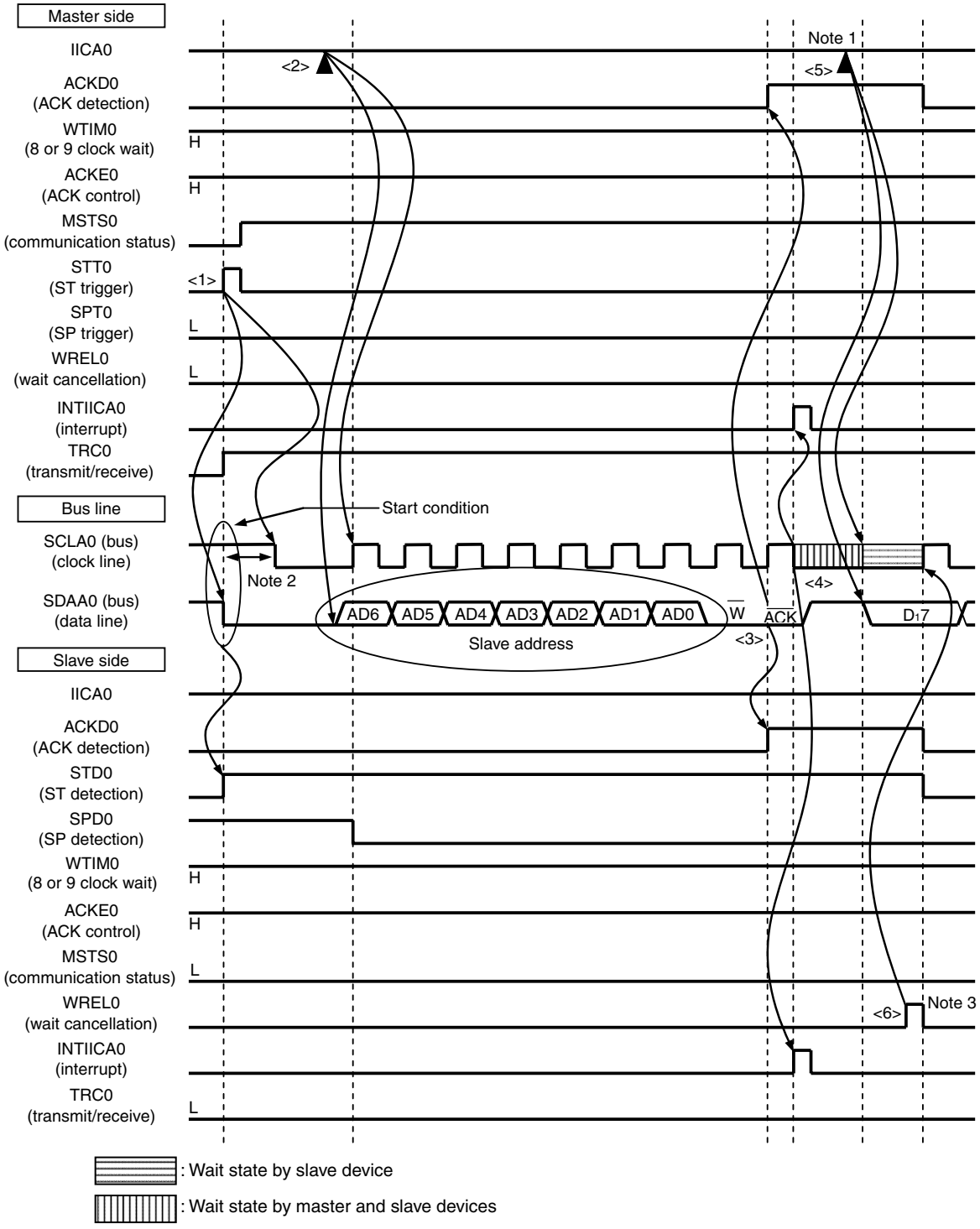
Figures 13-32 and 13-33 show timing charts of the data communication.

The IICA shift register 0 (IICA0)'s shift operation is synchronized with the falling edge of the serial clock (SCLA0). The transmit data is transferred to the SO latch and is output (MSB first) via the SDAA0 pin.

Data input via the SDAA0 pin is captured into IICA0 at the rising edge of SCLA0.

**Figure 13-32. Example of Master to Slave Communication (9-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (1/4)**

**(1) Start condition ~ address ~ data**




**Notes 1.** Write data to IICA0, not setting the WREL0 bit, in order to cancel a wait state during transmission by a master device.

**2.** Make sure that the time between the fall of the SDAA0 pin signal and the fall of the SCLA0 pin signal is at least 4.0 μs when specifying standard mode and at least 0.6 μs when specifying fast mode.

**3.** For releasing wait state during reception of a slave device, write “FFH” to IICA0 or set the WREL0 bit.

The meanings of <1> to <6> in (1) Start condition ~ address ~ data in Figure 13-32 are explained below.

<1> The start condition trigger is set by the master device (STT0 = 1) and a start condition (i.e. SCLA0 = 1 changes SDAA0 from 1 to 0) is generated once the bus data line goes low (SDAA0). When the start condition is subsequently detected, the master device enters the master device communication status (MSTS0 = 1). The master device is ready to communicate once the bus clock line goes low (SCLA0 = 0) after the hold time has elapsed.

<2> The master device writes the address + W (transmission) to the IICA shift register 0 (IICA0) and transmits the slave address.

<3> In the slave device if the address received matches the address (SVA0 value) of a slave device[Note], that slave device sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKD0 = 1) at the rising edge of the 9th clock.

<4> The master device issues an interrupt (INTIICA0: end of address transmission) at the falling edge of the 9th clock. The slave device whose address matched the transmitted slave address sets a wait status (SCLA0 = 0) and issues an interrupt (INTIICA0: address match)[Note].

<5> The master device writes the data to transmit to the IICA0 register and releases the wait status that it set by the master device.

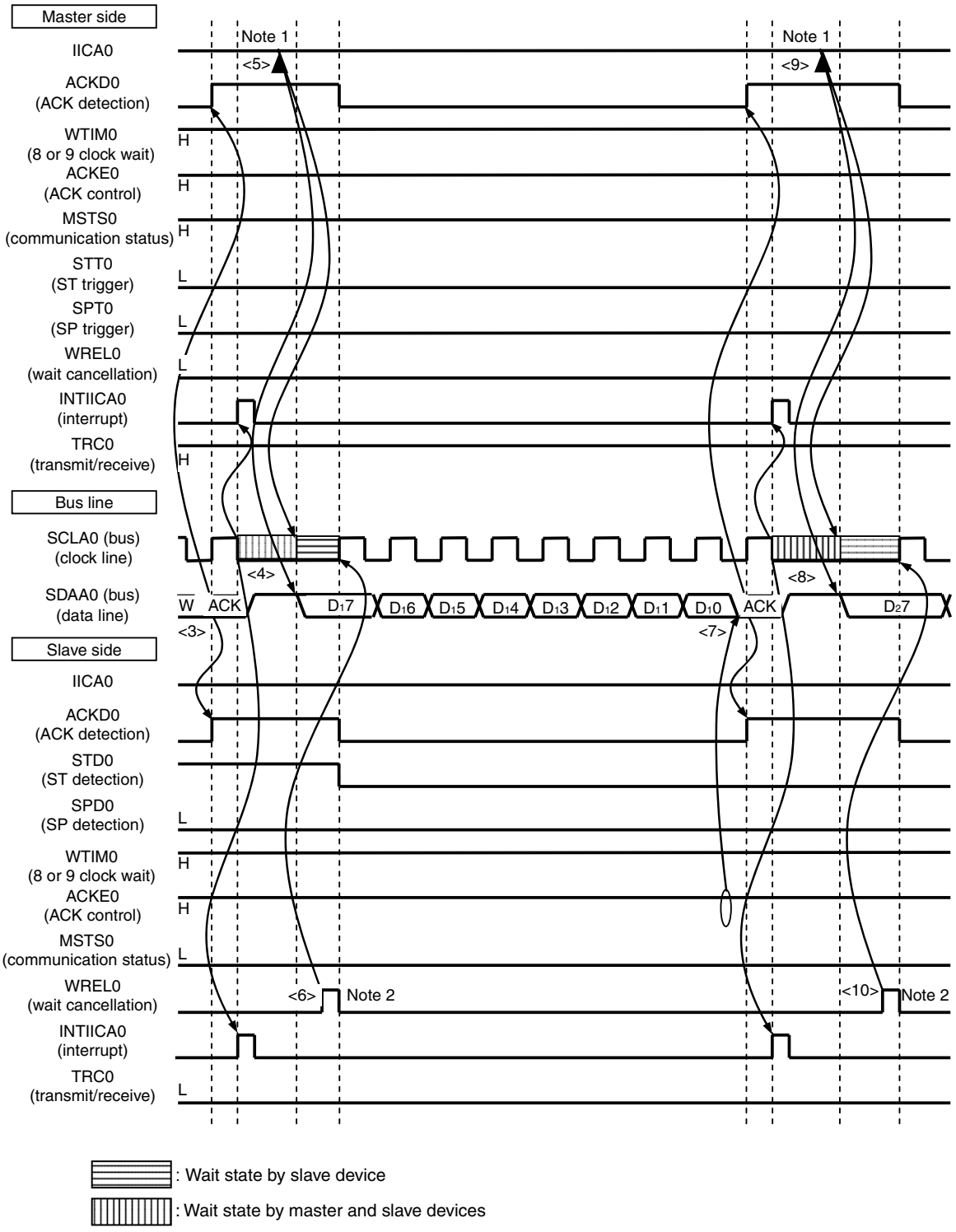
<6> If the slave device releases the wait status (WREL0 = 1), the master device starts transferring data to the slave device.

**Note** If the transmitted address does not match the address of the slave device, the slave device does not return an ACK to the master device (NACK: SDAA0 = 1). The slave device also does not issue the INTIICA0 interrupt (address match) and does not set a wait status. The master device, however, issues the INTIICA0 interrupt (end of address transmission) regardless of whether it receives an ACK or NACK.

**Remark** <1> to <15> in Figure 13-32 represent the entire procedure for communicating data using the $I^2C$ bus. Figure 13-32 (1) Start condition ~ address ~ data shows the processing from <1> to <6>, Figure 13-32 (2) Address ~ data ~ data shows the processing from <3> to <10>, and Figure 13-32 (3) Data ~ data ~ stop condition shows the processing from <7> to <15>.

**Figure 13-32. Example of Master to Slave Communication**
**(9-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (2/4)**

**(2) Address ~ data ~ data**




**Notes 1.** Write data to IICA0, not setting the WREL0 bit, in order to cancel a wait state during transmission by a master device.

**2.** For releasing wait state during reception of a slave device, write “FFH” to IICA0 or set the WREL0 bit.

The meanings of <3> to <10> in (2) Address ~ data ~ data in Figure 13-32 are explained below.

<3> In the slave device if the address received matches the address (SVA0 value) of a slave device[Note], that slave device sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKD0 = 1) at the rising edge of the 9th clock.

<4> The master device issues an interrupt (INTIICA0: end of address transmission) at the falling edge of the 9th clock. The slave device whose address matched the transmitted slave address sets a wait status (SCLA0 = 0) and issues an interrupt (INTIICA0: address match)[Note].

<5> The master device writes the data to transmit to the IICA shift register 0 (IICA0) and releases the wait status that it set by the master device.

<6> If the slave device releases the wait status (WREL0 = 1), the master device starts transferring data to the slave device.

<7> After data transfer is completed, because of ACKE0 = 1, the slave device sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKD0 = 1) at the rising edge of the 9th clock.

<8> The master device and slave device set a wait status (SCLA0 = 0) at the falling edge of the 9th clock, and both the master device and slave device issue an interrupt (INTIICA0: end of transfer).

<9> The master device writes the data to transmit to the IICA0 register and releases the wait status that it set by the master device.

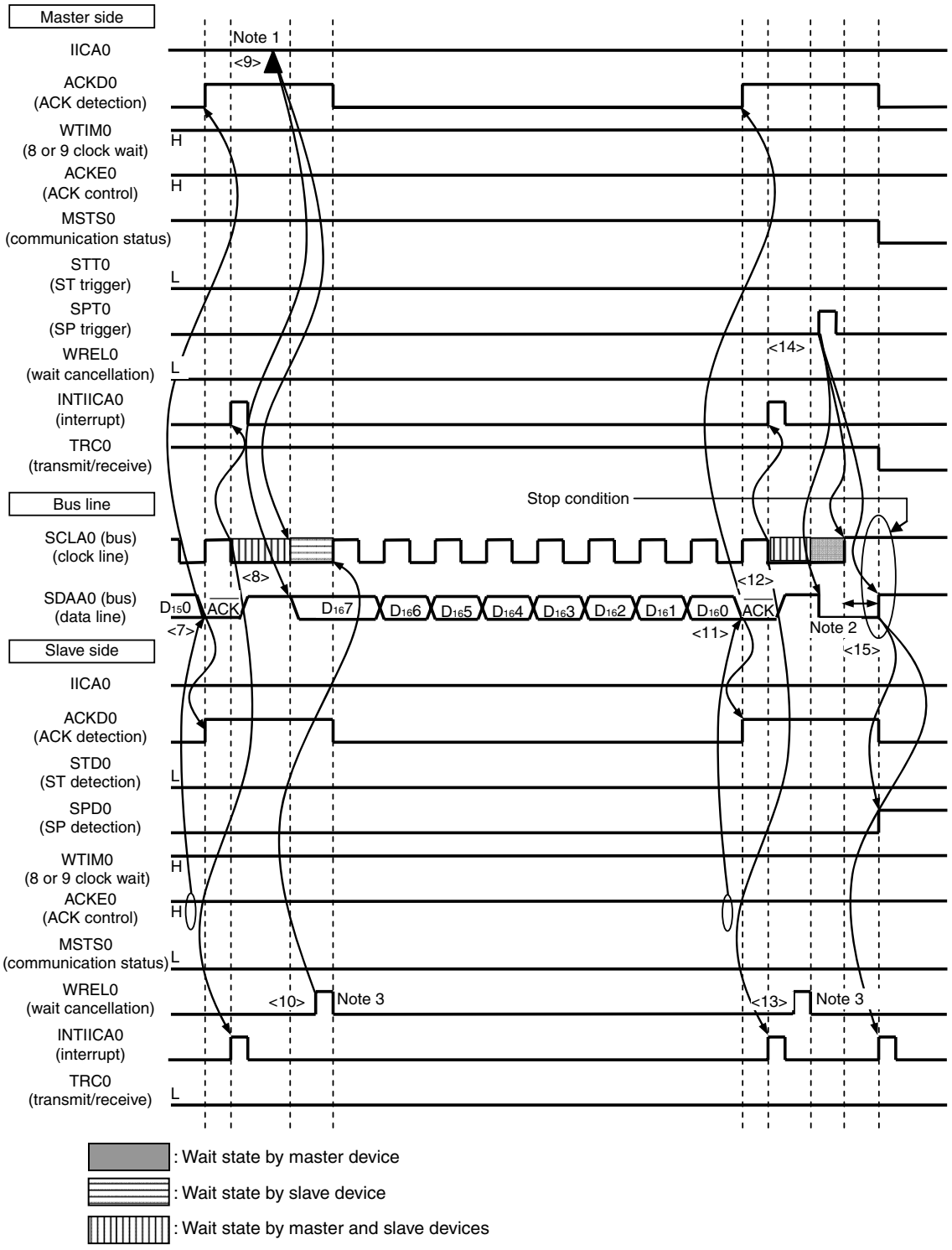
<10> The slave device reads the received data and releases the wait status (WREL0 = 1). The master device then starts transferring data to the slave device.

**Note** If the transmitted address does not match the address of the slave device, the slave device does not return an ACK to the master device (NACK: SDAA0 = 1). The slave device also does not issue the INTIICA0 interrupt (address match) and does not set a wait status. The master device, however, issues the INTIICA0 interrupt (end of address transmission) regardless of whether it receives an ACK or NACK.

**Remark** <1> to <15> in Figure 13-32 represent the entire procedure for communicating data using the $I^2C$ bus. Figure 13-32 (1) Start condition ~ address ~ data shows the processing from <1> to <6>, Figure 13-32 (2) Address ~ data ~ data shows the processing from <3> to <10>, and Figure 13-32 (3) Data ~ data ~ stop condition shows the processing from <7> to <15>.

**Figure 13-32. Example of Master to Slave Communication (9-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (3/4)**

**(3) Data ~ data ~ Stop condition**




**Notes 1.** Write data to IICA0, not setting the WREL0 bit, in order to cancel a wait state during transmission by a master device.

**2.** Make sure that the time between the rise of the SCLA0 pin signal and the generation of the stop condition after a stop condition has been issued is at least 4.0 μs when specifying standard mode and at least 0.6 μs when specifying fast mode.

**3.** For releasing wait state during reception of a slave device, write “FFH” to IICA0 or set the WREL0 bit.

The meanings of <7> to <15> in (3) Data ~ data ~ stop condition in Figure 13-32 are explained below.

<7> After data transfer is completed, because of ACKE0 = 1, the slave device sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKD0 = 1) at the rising edge of the 9th clock.

<8> The master device and slave device set a wait status (SCLA0 = 0) at the falling edge of the 9th clock, and both the master device and slave device issue an interrupt (INTIICA0: end of transfer).

<9> The master device writes the data to transmit to the IICA shift register 0 (IICA0) and releases the wait status that it set by the master device.

<10> The slave device reads the received data and releases the wait status (WREL0 = 1). The master device then starts transferring data to the slave device.

<11> When data transfer is complete, the slave device (ACKE0 =1) sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKD0 = 1) at the rising edge of the 9th clock.

<12> The master device and slave device set a wait status (SCLA0 = 0) at the falling edge of the 9th clock, and both the master device and slave device issue an interrupt (INTIICA0: end of transfer).

<13> The slave device reads the received data and releases the wait status (WREL0 = 1).

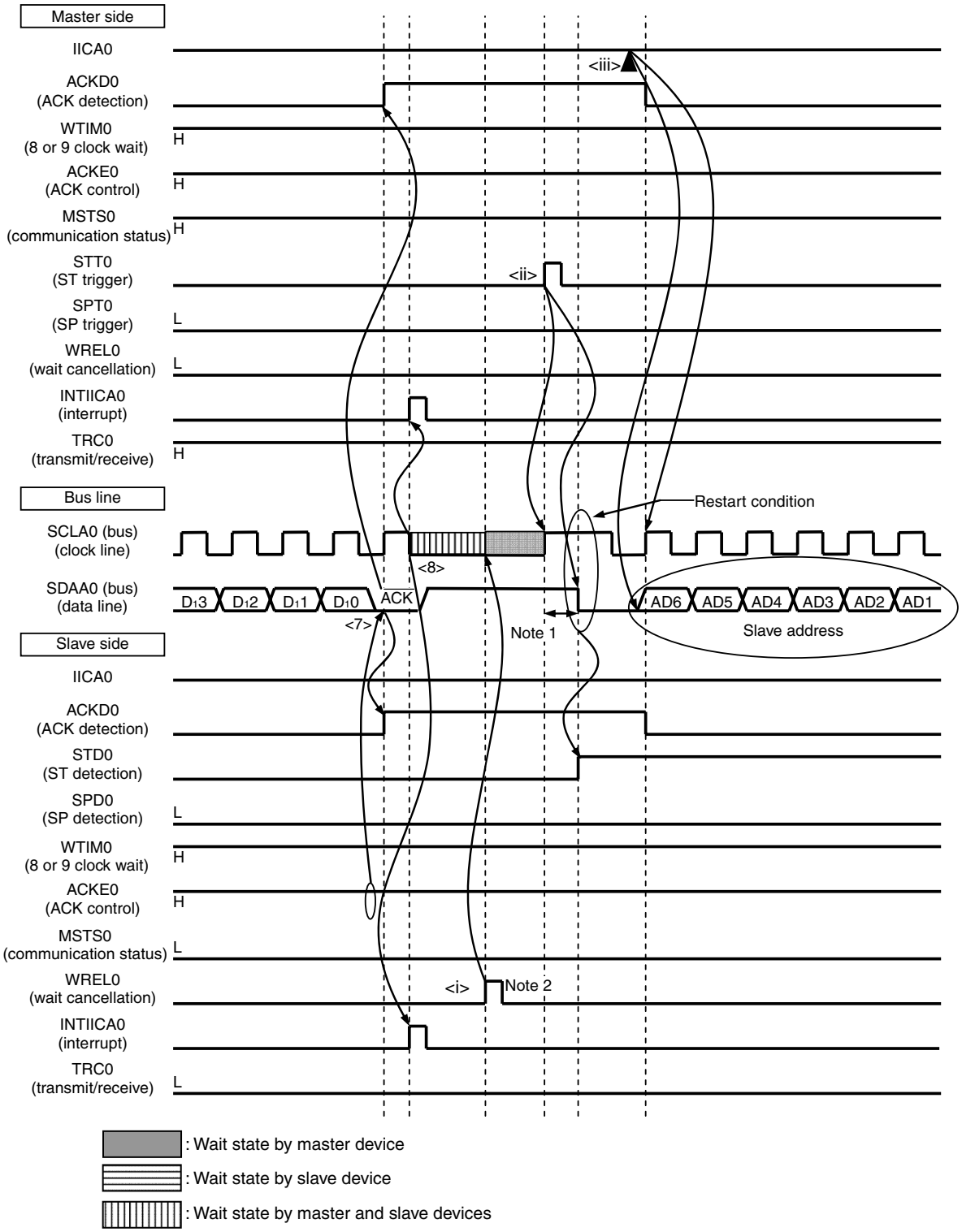
<14> By the master device setting a stop condition trigger (SPT0 = 1), the bus data line is cleared (SDAA0 = 0) and the bus clock line is set (SCLA0 = 1). After the stop condition setup time has elapsed, by setting the bus data line (SDAA0 = 1), the stop condition is then generated (i.e. SCLA0 =1 changes SDAA0 from 0 to 1).

<15> When a stop condition is generated, the slave device detects the stop condition and issues an interrupt (INTIICA0: stop condition).

**Remark** <1> to <15> in Figure 13-32 represent the entire procedure for communicating data using the $I^2C$ bus. Figure 13-32 (1) Start condition ~ address ~ data shows the processing from <1> to <6>, Figure 13-32 (2) Address ~ data ~ data shows the processing from <3> to <10>, and Figure 13-32 (3) Data ~ data ~ stop condition shows the processing from <7> to <15>.

**Figure 13-32. Example of Master to Slave Communication (9-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (4/4)**

**(4) Data ~ restart condition ~ address**




**Notes 1.** Make sure that the time between the rise of the SCLA0 pin signal and the generation of the start condition after a restart condition has been issued is at least 4.7 $\mu$s when specifying standard mode and at least 0.6 $\mu$s when specifying fast mode.

**2.** For releasing wait state during reception of a slave device, write “FFH” to IICA0 or set the WREL0 bit.

The following describes the operations in Figure 13-32 (4) Data ~ restart condition ~ address. After the operations in steps <7> and <8>, the operations in steps <i> to <iii> are performed. These steps return the processing to step <iii>, the data transmission step.

<7> After data transfer is completed, because of ACKE0 = 1, the slave device sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKD0 = 1) at the rising edge of the 9th clock.

<8> The master device and slave device set a wait status (SCLA0 = 0) at the falling edge of the 9th clock, and both the master device and slave device issue an interrupt (INTIICA0: end of transfer).

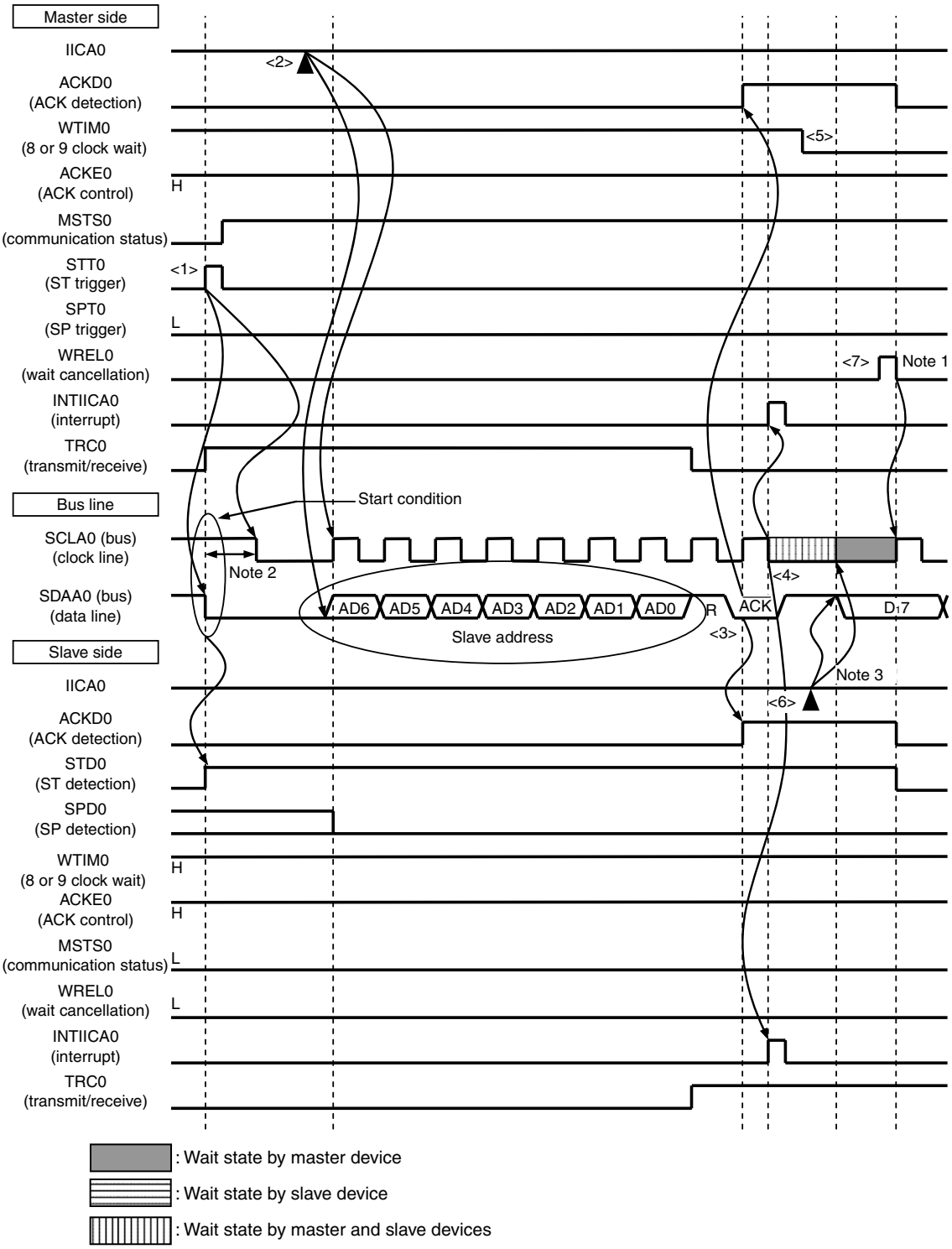
<i> The slave device reads the received data and releases the wait status (WREL0 = 1).

<ii> The start condition trigger is set again by the master device (STT0 = 1) and a start condition (i.e. SCLA0 =1 changes SDAA0 from 1 to 0) is generated once the bus clock line goes high (SCLA0 = 1) and the bus data line goes low (SDAA0 = 0) after the restart condition setup time has elapsed. When the start condition is subsequently detected, the master device is ready to communicate once the bus clock line goes low (SCLA0 = 0) after the hold time has elapsed.

<iii> The master device writing the address + R/W (transmission) to the IICA shift register (IICA0) enables the slave address to be transmitted.

**Figure 13-33. Example of Slave to Master Communication (8-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (1/3)**

**(1) Start condition ~ address ~ data**




**Notes 1.** For releasing wait state during reception of a master device, write “FFH” to IICA0 or set the WREL0 bit.

**2.** Make sure that the time between the fall of the SDAA0 pin signal and the fall of the SCLA0 pin signal is at least 4.0 $\mu$s when specifying standard mode and at least 0.6 $\mu$s when specifying fast mode.

**3.** Write data to IICA0, not setting the WREL0 bit, in order to cancel a wait state during transmission by a slave device.

The meanings of <1> to <7> in (1) Start condition ~ address ~ data in Figure 13-33 are explained below.

<1> The start condition trigger is set by the master device (STT0 = 1) and a start condition (i.e. SCLA0 =1 changes SDAA0 from 1 to 0) is generated once the bus data line goes low (SDAA0). When the start condition is subsequently detected, the master device enters the master device communication status (MSTS0 = 1). The master device is ready to communicate once the bus clock line goes low (SCLA0 = 0) after the hold time has elapsed.

<2> The master device writes the address + R (reception) to the IICA shift register 0 (IICA0) and transmits the slave address.

<3> In the slave device if the address received matches the address (SVA0 value) of a slave device[Note], that slave device sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKD0 = 1) at the rising edge of the 9th clock.

<4> The master device issues an interrupt (INTIICA0: end of address transmission) at the falling edge of the 9th clock. The slave device whose address matched the transmitted slave address sets a wait status (SCLA0 = 0) and issues an interrupt (INTIICA0: address match)[Note].

<5> The timing at which the master device sets the wait status changes to the 8th clock (WTIM0 = 0).

<6> The slave device writes the data to transmit to the IICA0 register and releases the wait status that it set by the slave device.

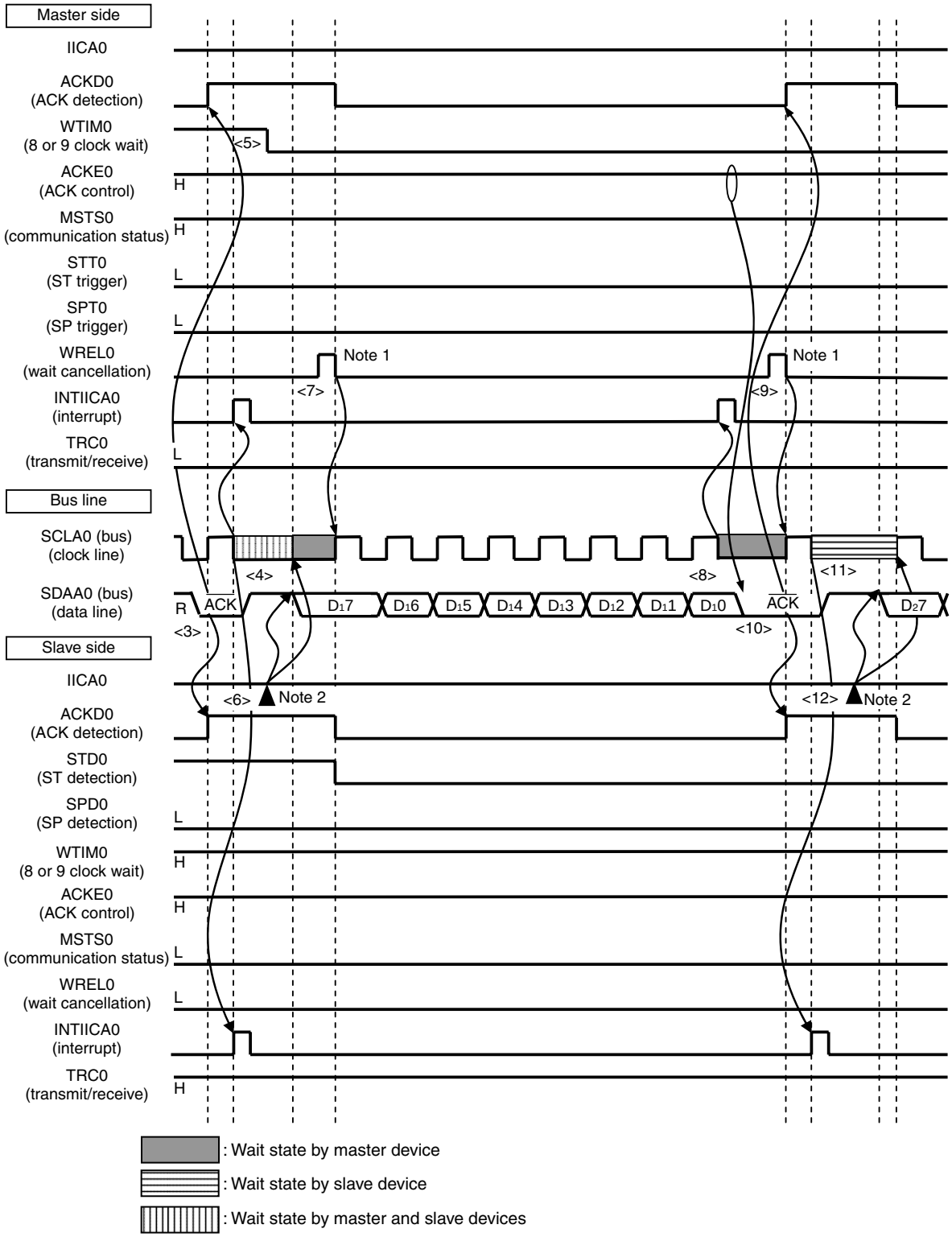
<7> The master device releases the wait status (WREL0 = 1) and starts transferring data from the slave device to the master device.

**Note** If the transmitted address does not match the address of the slave device, the slave device does not return an ACK to the master device (NACK: SDAA0 = 1). The slave device also does not issue the INTIICA0 interrupt (address match) and does not set a wait status. The master device, however, issues the INTIICA0 interrupt (end of address transmission) regardless of whether it receives an ACK or NACK.

**Remark** <1> to <19> in Figure 13-33 represent the entire procedure for communicating data using the $I^2C$ bus. Figure 13-33 (1) Start condition ~ address ~ data shows the processing from <1> to <7>, Figure 13-33 (2) Address ~ data ~ data shows the processing from <3> to <12>, and Figure 13-33 (3) Data ~ data ~ stop condition shows the processing from <8> to <19>.

**Figure 13-33. Example of Slave to Master Communication (8-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (2/3)**

**(2) Address ~ data ~ data**




**Notes 1.** For releasing wait state during reception of a master device, write “FFH” to IICA0 or set the WREL0 bit.

**2.** Write data to IICA0, not setting the WREL0 bit, in order to cancel a wait state during transmission by a slave device.

The meanings of <3> to <12> in (2) Address ~ data ~ data in Figure 13-33 are explained below.

<3> In the slave device if the address received matches the address (SVA0 value) of a slave device[Note], that slave device sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKD0 = 1) at the rising edge of the 9th clock.

<4> The master device issues an interrupt (INTIICA0: end of address transmission) at the falling edge of the 9th clock. The slave device whose address matched the transmitted slave address sets a wait status (SCLA0 = 0) and issues an interrupt (INTIICA0: address match)[Note].

<5> The master device changes the timing of the wait status to the 8th clock (WTIM0 = 0).

<6> The slave device writes the data to transmit to the IICA shift register 0 (IICA0) and releases the wait status that it set by the slave device.

<7> The master device releases the wait status (WREL0 = 1) and starts transferring data from the slave device to the master device.

<8> The master device sets a wait status (SCLA0 = 0) at the falling edge of the 8th clock, and issues an interrupt (INTIICA0: end of transfer). Because of ACKE0 = 1 in the master device, the master device then sends an ACK by hardware to the slave device.

<9> The master device reads the received data and releases the wait status (WREL0 = 1).

<10> The ACK is detected by the slave device (ACKD0 = 1) at the rising edge of the 9th clock.

<11> The slave device set a wait status (SCLA0 = 0) at the falling edge of the 9th clock, and the slave device issue an interrupt (INTIICA0: end of transfer).

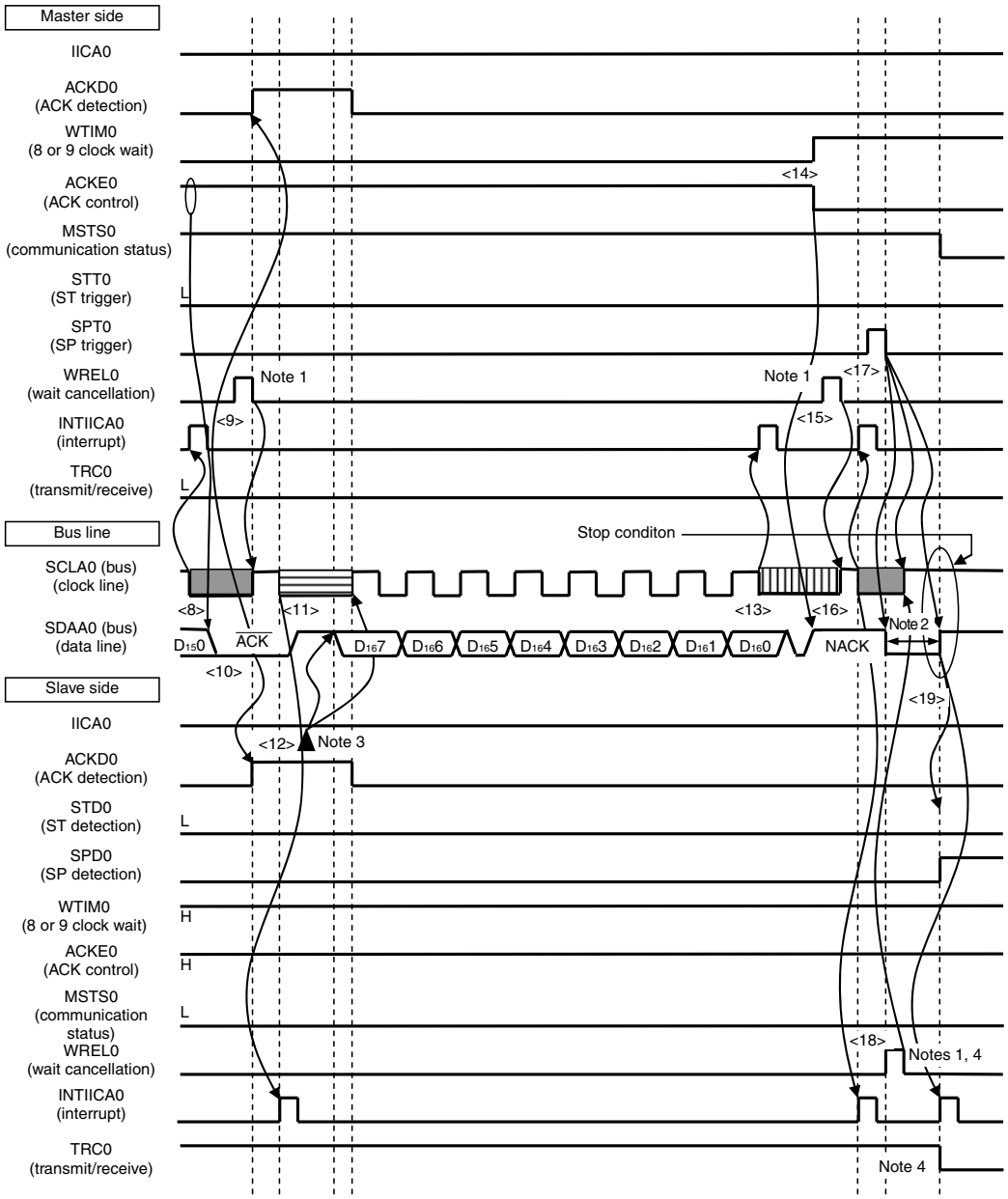
<12> By the slave device writing the data to transmit to the IICA0 register, the wait status set by the slave device is released. The slave device then starts transferring data to the master device.

**Note** If the transmitted address does not match the address of the slave device, the slave device does not return an ACK to the master device (NACK: SDAA0 = 1). The slave device also does not issue the INTIICA0 interrupt (address match) and does not set a wait status. The master device, however, issues the INTIICA0 interrupt (end of address transmission) regardless of whether it receives an ACK or NACK.

**Remark** <1> to <19> in Figure 13-33 represent the entire procedure for communicating data using the $I^2C$ bus. Figure 13-33 (1) Start condition ~ address ~ data shows the processing from <1> to <7>, Figure 13-33 (2) Address ~ data ~ data shows the processing from <3> to <12>, and Figure 13-33 (3) Data ~ data ~ stop condition shows the processing from <8> to <19>.

**Figure 13-33. Example of Slave to Master Communication (8-Clock and 9-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (3/3)**

**(3) Data ~ data ~ stop condition**




**Notes 1.** To cancel a wait state, write “FFH” to IICA0 or set the WREL0 bit.

**2.** Make sure that the time between the rise of the SCLA0 pin signal and the generation of the stop condition after a stop condition has been issued is at least 4.0 $\mu$s when specifying standard mode and at least 0.6 $\mu$s when specifying fast mode.

**3.** Write data to IICA0, not setting the WREL0 bit, in order to cancel a wait state during transmission by a slave device.

**4.** If a wait state during transmission by a slave device is canceled by setting the WREL0 bit, the TRC0 bit will be cleared.

The meanings of <8> to <19> in (3) Data ~ data ~ stop condition in Figure 13-33 are explained below.

<8> The master device sets a wait status (SCLA0 = 0) at the falling edge of the 8th clock, and issues an interrupt (INTIICA0: end of transfer). Because of ACKE0 = 0 in the master device, the master device then sends an ACK by hardware to the slave device.

<9> The master device reads the received data and releases the wait status (WREL0 = 1).

<10> The ACK is detected by the slave device (ACKD0 = 1) at the rising edge of the 9th clock.

<11> The slave device set a wait status (SCLA0 = 0) at the falling edge of the 9th clock, and the slave device issue an interrupt (INTIICA0: end of transfer).

<12> By the slave device writing the data to transmit to the IICA register, the wait status set by the slave device is released. The slave device then starts transferring data to the master device.

<13> The master device issues an interrupt (INTIICA0: end of transfer) at the falling edge of the 8th clock, and sets a wait status (SCLA0 = 0). Because ACK control (ACKE0 = 1) is performed, the bus data line is at the low level (SDAA0 = 0) at this stage.

<14> The master device sets NACK as the response (ACKE0 = 0) and changes the timing at which it sets the wait status to the 9th clock (WTIM0 = 1).

<15> If the master device releases the wait status (WREL0 = 1), the slave device detects the NACK (ACK = 0) at the rising edge of the 9th clock.

<16> The master device and slave device set a wait status (SCLA0 = 0) at the falling edge of the 9th clock, and both the master device and slave device issue an interrupt (INTIICA0: end of transfer).

<17> When the master device issues a stop condition (SPT0 = 1), the bus data line is cleared (SDAA0 = 0) and the master device releases the wait status. The master device then waits until the bus clock line is set (SCLA0 = 1).

<18> The slave device acknowledges the NACK, halts transmission, and releases the wait status (WREL0 = 1) to end communication. Once the slave device releases the wait status, the bus clock line is set (SCLA0 = 1).

<19> Once the master device recognizes that the bus clock line is set (SCLA0 = 1) and after the stop condition setup time has elapsed, the master device sets the bus data line (SDAA0 = 1) and issues a stop condition (i.e. SCLA0 =1 changes SDAA0 from 0 to 1). The slave device detects the generated stop condition and slave device issue an interrupt (INTIICA0: stop condition).

**Remark** <1> to <19> in Figure 13-33 represent the entire procedure for communicating data using the $I^2C$ bus. Figure 13-33 (1) Start condition ~ address ~ data shows the processing from <1> to <7>, Figure 13-33 (2) Address ~ data ~ data shows the processing from <3> to <12>, and Figure 13-33 (3) Data ~ data ~ stop condition shows the processing from <8> to <19>.

# CHAPTER 14 MULTIPLIER AND DIVIDER/MULTIPLY-ACCUMULATOR

## 14.1 Functions of Multiplier and Divider/Multiply-Accumulator

The multiplier and divider/multiply-accumulator has the following functions.

- 16 bits × 16 bits = 32 bits (Unsigned)
- 16 bits × 16 bits = 32 bits (Signed)
- 16 bits × 16 bits + 32 bits = 32 bits (Unsigned)
- 16 bits × 16 bits + 32 bits = 32 bits (Signed)
- 32 bits ÷ 32 bits = 32 bits, 32-bits remainder (Unsigned)

## 14.2 Configuration of Multiplier and Divider/Multiply-Accumulator

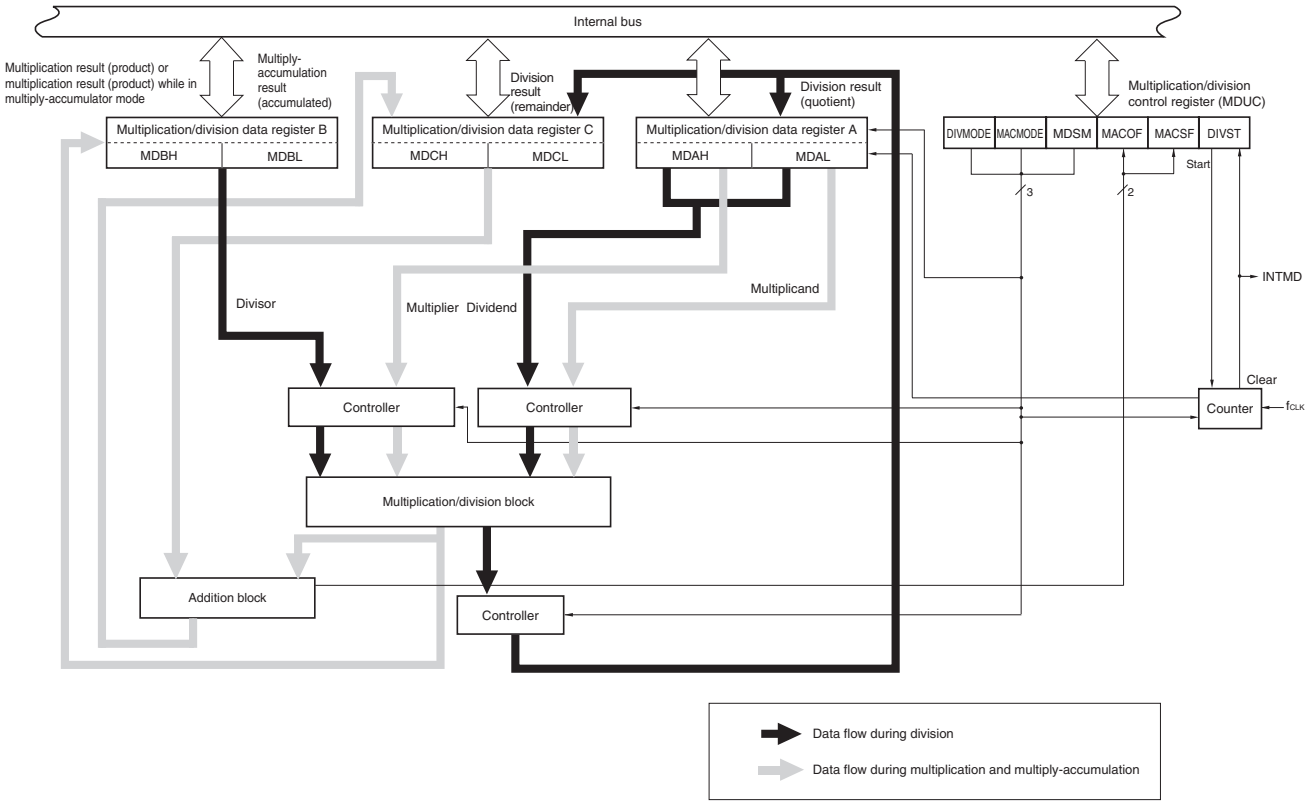
The multiplier and divider/multiply-accumulator consists of the following hardware.

**Table 14-1. Configuration of Multiplier and Divider/Multiply-Accumulator**

| Item | Configuration |
|---|---|
| Registers | Multiplication/division data register A (L) (MDAL)<br>Multiplication/division data register A (H) (MDAH)<br>Multiplication/division data register B (L) (MDBL)<br>Multiplication/division data register B (H) (MDBH)<br>Multiplication/division data register C (L) (MDCL)<br>Multiplication/division data register C (H) (MDCH) |
| Control register | Multiplication/division control register (MDUC) |

Figure 14-1 shows a block diagram of the multiplier and divider/multiply-accumulator.

<R> **Figure 14-1. Block Diagram of Multiplier and Divider/Multiply-Accumulator**




<R> **Remark** $f_{CLK}$: CPU/peripheral hardware clock frequency

### 14.2.1 Multiplication/division data register A (MDAH, MDAL)

The MDAH and MDAL registers set the values that are used for a multiplication or division operation and store the operation result. They set the multiplier and multiplicand data in the multiplication mode or multiply-accumulator mode, and set the dividend data in the division mode. Furthermore, the operation result (quotient) is stored in the MDAH and MDAL registers in the division mode.

The MDAH and MDAL registers can be set by a 16-bit manipulation instruction.

Reset signal generation clears these registers to 0000H.

**Figure 14-2. Format of Multiplication/Division Data Register A (MDAH, MDAL)**

Address: FFFF0H, FFFF1H, FFFF2H, FFFF3H After reset: 0000H, 0000H R/W

Symbol FFFF3H FFFF2H

| MDAH | MDAH 15 | MDAH 14 | MDAH 13 | MDAH 12 | MDAH 11 | MDAH 10 | MDAH 9 | MDAH 8 | MDAH 7 | MDAH 6 | MDAH 5 | MDAH 4 | MDAH 3 | MDAH 2 | MDAH 1 | MDAH 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

Symbol FFFF1H FFFF0H

| MDAL | MDAL 15 | MDAL 14 | MDAL 13 | MDAL 12 | MDAL 11 | MDAL 10 | MDAL 9 | MDAL 8 | MDAL 7 | MDAL 6 | MDAL 5 | MDAL 4 | MDAL 3 | MDAL 2 | MDAL 1 | MDAL 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

**Cautions 1. Do not rewrite the MDAH and MDAL registers values during division operation processing (when the multiplication/division control register (MDUC) value is 81H or C1H). The operation will be executed in this case, but the operation result will be an undefined value.**

**2. The MDAH and MDAL registers values read during division operation processing (when the MDUC register value is 81H or C1H) will not be guaranteed.**

**3. The data is in the two's complement format in either the multiplication mode (signed) or multiply-accumulator mode (signed).**

The following table shows the functions of the MDAH and MDAL registers during operation execution.

**Table 14-2. Functions of MDAH and MDAL Registers During Operation Execution**

| Operation Mode | Setting | Operation Result |
|---|---|---|
| Multiplication mode (unsigned)<br>Multiply-accumulator mode (unsigned) | MDAH: Multiplier (unsigned)<br>MDAL: Multiplicand (unsigned) | – |
| Multiplication mode (signed)<br>Multiply-accumulator mode (signed) | MDAH: Multiplier (signed)<br>MDAL: Multiplicand (signed) | – |
| Division mode (unsigned) | MDAH: Dividend (unsigned) (higher 16 bits)<br>MDAL: Dividend (unsigned) (lower 16 bits) | MDAH: Division result (unsigned) Higher 16 bits<br>MDAL: Division result (unsigned Lower 16 bits |

### 14.2.2 Multiplication/division data register B (MDBL, MDBH)

The MDBH and MDBL registers set the values that are used for multiplication or division operation and store the operation result. They store the operation result (product) in the multiplication mode and multiply-accumulator mode, and set the divisor data in the division mode.

The MDBH and MDBL registers can be set by a 16-bit manipulation instruction.

Reset signal generation clears these registers to 0000H.

**Figure 14-3. Format of Multiplication/Division Data Register B (MDBH, MDBL)**

Address: FFFF4H, FFFF5H, FFFF6H, FFFF7H After reset: 0000H, 0000H R/W

| Symbol | FFFF5H | | | | | | | | FFFF4H | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MDBH | MDBH 15 | MDBH 14 | MDBH 13 | MDBH 12 | MDBH 11 | MDBH 10 | MDBH 9 | MDBH 8 | MDBH 7 | MDBH 6 | MDBH 5 | MDBH 4 | MDBH 3 | MDBH 2 | MDBH 1 | MDBH 0 |

| Symbol | FFFF7H | | | | | | | | FFFF6H | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MDBL | MDBL 15 | MDBL 14 | MDBL 13 | MDBL 12 | MDBL 11 | MDBL 10 | MDBL 9 | MDBL 8 | MDBL 7 | MDBL 6 | MDBL 5 | MDBL 4 | MDBHL 3 | MDBL 2 | MDBL 1 | MDBL 0 |

**Cautions 1. Do not rewrite the MDBH and MDBL registers values during division operation processing (when the multiplication/division control register (MDUC) value is 81H or C1H) or multiply-accumulation operation processing. The operation result will be an undefined value.**

**2. Do not set the MDBH and MDBL registers to 0000H in the division mode. If they are set, the operation result will be an undefined value.**

**3. The data is in the two's complement format in either the multiplication mode (signed) or multiply-accumulator mode (signed).**

The following table shows the functions of the MDBH and MDBL registers during operation execution.

**Table 14-3. Functions of MDBH and MDBL Registers During Operation Execution**

| Operation Mode | Setting | Operation Result |
|---|---|---|
| Multiplication mode (unsigned)<br>Multiply-accumulator mode (unsigned) | – | MDBH: Multiplication result (product) (unsigned) Higher 16 bits<br>MDBL: Multiplication result (product) (unsigned) Lower 16 bits |
| Multiplication mode (signed)<br>Multiply-accumulator mode (signed) | – | MDBH: Multiplication result (product) (signed) Higher 16 bits<br>MDBL: Multiplication result (product) (signed) Lower 16 bits |
| Division mode (unsigned) | MDBH: Divisor (unsigned) (higher 16 bits)<br>MDBL: Divisor (unsigned) (lower 16 bits) | – |

### 14.2.3 Multiplication/division data register C (MDCL, MDCH)

The MDCH and MDCL registers are used to store the accumulated result while in the multiply-accumulator mode or the remainder of the operation result while in the division mode. These registers are not used while in the multiplication mode.

The MDCH and MDCL registers can be set by a 16-bit manipulation instruction.

Reset signal generation clears these registers to 0000H.

**Figure 14-4. Format of Multiplication/Division Data Register C (MDCH, MDCL)**

Address: F00E0H, F00E1H, F00E2H, F00E3H After reset: 0000H, 0000H R/W

Symbol F00E3H F00E2H

| MDCH | MDCH 15 | MDCH 14 | MDCH 13 | MDCH 12 | MDCH 11 | MDCH 10 | MDCH 9 | MDCH 8 | MDCH 7 | MDCH 6 | MDCH 5 | MDCH 4 | MDCH 3 | MDCH 2 | MDCH 1 | MDCH 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

Symbol F00E1H F00E0H

| MDCL | MDCL 15 | MDCL 14 | MDCL 13 | MDCL 12 | MDCL 11 | MDCL 10 | MDCL 9 | MDCL 8 | MDCL 7 | MDCL 6 | MDCL 5 | MDCL 4 | MDCL 3 | MDCL 2 | MDCL 1 | MDCL 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

**Cautions 1. The MDCH and MDCL registers values read during division operation processing (when the multiplication/division control register (MDUC) value is 81H or C1H) will not be guaranteed.**

**2. During multiply-accumulator processing, do not use software to rewrite the values of the MDCH and MDCL registers. If this is done, the operation result will be undefined.**

**3. The data is in the two's complement format in the multiply-accumulator mode (signed).**

**Table 14-4. Functions of MDCH and MDCL Registers During Operation Execution**

| Operation Mode | Setting | Operation Result |
|---|---|---|
| Multiplication mode (unsigned or signed) | – | – |
| Multiply-accumulator mode (unsigned) | MDCH: Initial accumulated value (unsigned) (higher 16 bits)<br>MDCL: Initial accumulated value (unsigned) (lower 16 bits) | MDCH: accumulated value (unsigned) (higher 16 bits)<br>MDCL: accumulated value (unsigned) (lower 16 bits) |
| Multiply-accumulator mode (signed) | MDCH: Initial accumulated value (signed) (higher 16 bits)<br>MDCL: Initial accumulated value (signed) (lower 16 bits) | MDCH: accumulated value (signed) (higher 16 bits)<br>MDCL: accumulated value (signed) (lower 16 bits) |
| Division mode (unsigned) | – | MDCH: Remainder (unsigned) (higher 16 bits)<br>MDCL: Remainder (unsigned) (lower 16 bits) |

The register configuration differs between when multiplication is executed and when division is executed, as follows.

- Register configuration during multiplication

  <Multiplier A> <Multiplier B> <Product>

  MDAL (bits 15 to 0) × MDAH (bits 15 to 0) = [MDBH (bits 15 to 0), MDBL (bits 15 to 0)]

- Register configuration during multiply-accumulation

  <Multiplier A> <Multiplier B> < accumulated value > < accumulated result >

  MDAL (bits 15 to 0) × MDAH (bits 15 to 0) + MDC (bits 31 to 0) = [MDCH (bits 15 to 0), MDCL (bits 15 to 0)]

  (The multiplication result is stored in the MDBH (bits 15 to 0) and MDBL (bits 15 to 0).)

- Register configuration during division

  <Dividend> <Divisor>

  [MDAH (bits 15 to 0), MDAL (bits 15 to 0)] ÷ [MDBH (bits 15 to 0), MDBL (bits 15 to 0)] =

  <Quotient> <Remainder>

  [MDAH (bits 15 to 0), MDAL (bits 15 to 0)] ··· [MDCH (bits 15 to 0), MDCL (bits 15 to 0)]

## 14.3 Register Controlling Multiplier and Divider/Multiply-Accumulator

The multiplier and divider/multiply-accumulator is controlled by using the multiplication/division control register (MDUC).

### 14.3.1 Multiplication/division control register (MDUC)

The MDUC register is an 8-bit register that controls the operation of the multiplier and divider/multiply-accumulator.

The MDUC register can be set by a 1-bit or 8-bit memory manipulation instruction.

<R> Note that the overflow flag (MACOF) and sign flag (MACSF) of the multiply-accumulation result (accumulated) are read-only flags.

Reset signal generation clears this register to 00H.

**Figure 14-5. Format of Multiplication/Division Control Register (MDUC)**

Address: F00E8H After reset: 00H R/W[Note 1]

| Symbol | <7> | <6> | 5 | 4 | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| MDUC | DIVMODE | MACMODE | 0 | 0 | MDSM | MACOF | MACSF | DIVST |

| | DIVMODE | MACMODE | MDSM | Operation mode selection |
|---|---|---|---|---|
| | 0 | 0 | 0 | Multiplication mode (unsigned) (default) |
| | 0 | 0 | 1 | Multiplication mode (signed) |
| | 0 | 1 | 0 | Multiply-accumulator mode (unsigned) |
| | 0 | 1 | 1 | Multiply-accumulator mode (signed) |
| <R> | 1 | 0 | 0 | Division mode (unsigned), generation of a division completion interrupt (INTMD) |
| <R> | 1 | 1 | 0 | Division mode (unsigned), not generation of a division completion interrupt (INTMD) |
| | Other than above | | | Setting prohibited |

| MACOF | Overflow flag of multiply-accumulation result (accumulated value) |
|---|---|
| 0 | No overflow |
| 1 | With over flow |
| <Set condition><br>• For the multiply-accumulator mode (unsigned)<br>The bit is set when the accumulated value goes outside the range from 00000000h to FFFFFFFFh.<br>• For the multiply-accumulator mode (signed)<br>The bit is set when the result of adding a positive product to a positive accumulated value exceeds 7FFFFFFFh and is negative, or when the result of adding a negative product to a negative accumulated value exceeds 80000000h and is positive. | |

| MACSF | Sign flag of multiply-accumulation result (accumulated value) |
|---|---|
| 0 | The accumulated value is positive. |
| 1 | The accumulated value is negative. |
| Multiply-accumulator mode (unsigned): The bit is always 0.<br>Multiply-accumulator mode (signed): The bit indicates the sign bit of the accumulated value. | |

| DIVST[Note 2] | Division operation start/stop |
|---|---|
| 0 | Division operation processing complete |
| 1 | Starts division operation/division operation processing in progress |

(**Notes** and **Cautions** are listed on the next page.)

**Notes 1.** Bits 1 and 2 are read-only bits.

**2.** The DIVST bit can only be set (1) in the division mode. In the division mode, division operation is started by setting (1) the DIVST bit. The DIVST bit is automatically cleared (0) when the operation ends. In the multiplication mode, operation is automatically started by setting the multiplier and multiplicand to multiplication/division data register A (MDAH, MDAL), respectively.

**Cautions 1. Do not rewrite the DIVMODE, MDSM bits during operation processing (while the DIVST bit is 1). If it is rewritten, the operation result will be an undefined value.**

**2. The DIVST bit cannot be cleared (0) by using software during division operation processing (while the DIVST bit is 1).**

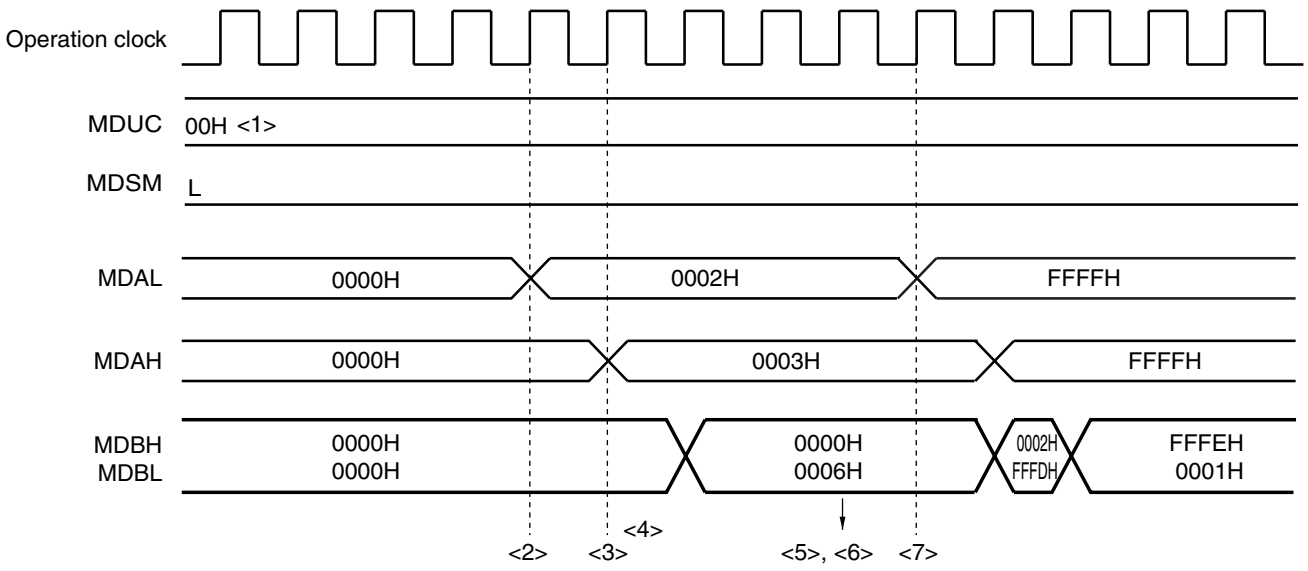
## 14.4 Operations of Multiplier and Divider/Multiply-Accumulator

### 14.4.1 Multiplication (unsigned) operation

- Initial setting
  - <1> Set the multiplication/division control register (MDUC) to 00H.
  - <2> Set the multiplicand to multiplication/division data register A (L) (MDAL).
  - <3> Set the multiplier to multiplication/division data register A (H) (MDAH).
    (There is no preference in the order of executing steps <2> and <3>. Multiplication operation is automatically started when the multiplier and multiplicand are set to the MDAH and MDAL registers, respectively.)
- During operation processing
  - <4> Wait for at least one clock. The operation will end when one clock has been issued.
- Operation end
  - <5> Read the product (lower 16 bits) from multiplication/division data register B (L) (MDBL).
  - <6> Read the product (higher 16 bits) from multiplication/division data register B (H) (MDBH).
    (There is no preference in the order of executing steps <5> and <6>.)
- Next operation
  - <7> Start with the initial settings of each step to change the operation mode. When the same operation mode is used sequentially, settings <1> and <2> can be omitted.

**Remark** Steps <1> to <7> correspond to <1> to <7> in Figure 14-6.

**Figure 14-6. Timing Diagram of Multiplication (Unsigned) Operation (2 × 3 = 6)**



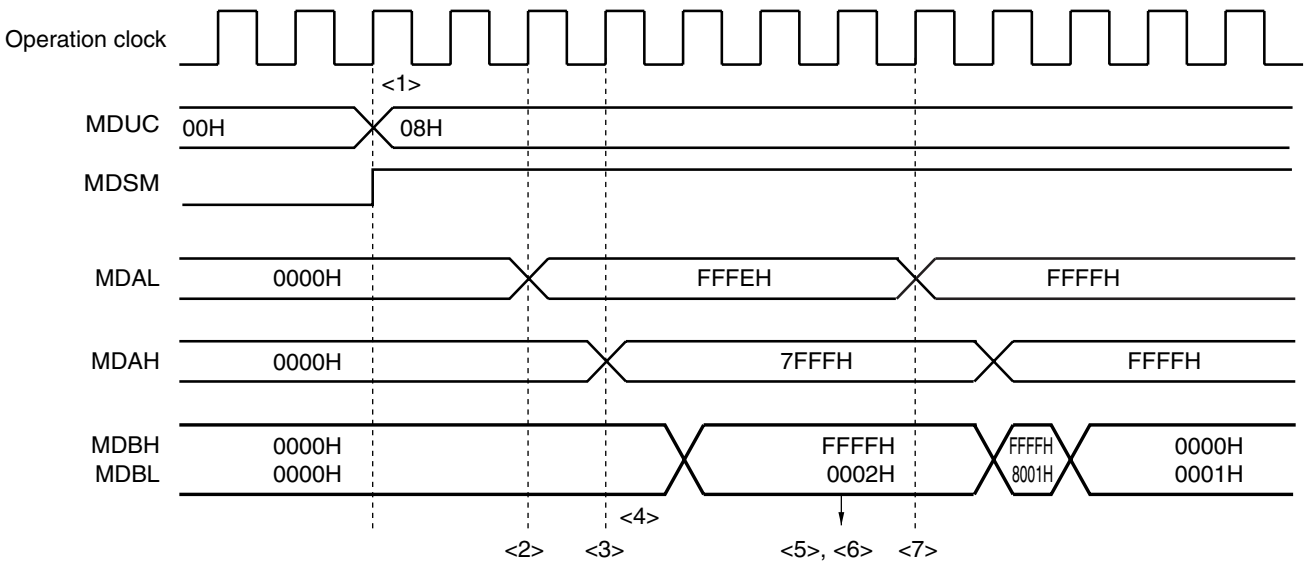
### 14.4.2 Multiplication (signed) operation

- Initial setting
  - <1> Set the multiplication/division control register (MDUC) to 08H.
  - <2> Set the multiplicand to multiplication/division data register A (L) (MDAL).
  - <3> Set the multiplier to multiplication/division data register A (H) (MDAH).
    (There is no preference in the order of executing steps <2> and <3>. Multiplication operation is automatically started when the multiplier and multiplicand are set to the MDAH and MDAL registers, respectively.)
- During operation processing
  - <4> Wait for at least one clock. The operation will end when one clock has been issued.
- Operation end
  - <5> Read the product (lower 16 bits) from multiplication/division data register B (L) (MDBL).
  - <6> Read the product (higher 16 bits) from multiplication/division data register B (H) (MDBH).
    (There is no preference in the order of executing steps <5> and <6>.)
- Next operation
  - <7> Start with the initial settings of each step to change the operation mode. When the same operation mode is used sequentially, settings <1> and <2> can be omitted.

**Caution The data is in the two's complement format in multiplication mode (signed).**

**Remark** Steps <1> to <7> correspond to <1> to <7> in Figure 14-7.

**Figure 14-7. Timing Diagram of Multiplication (Signed) Operation (–2 × 32767 = –65534)**

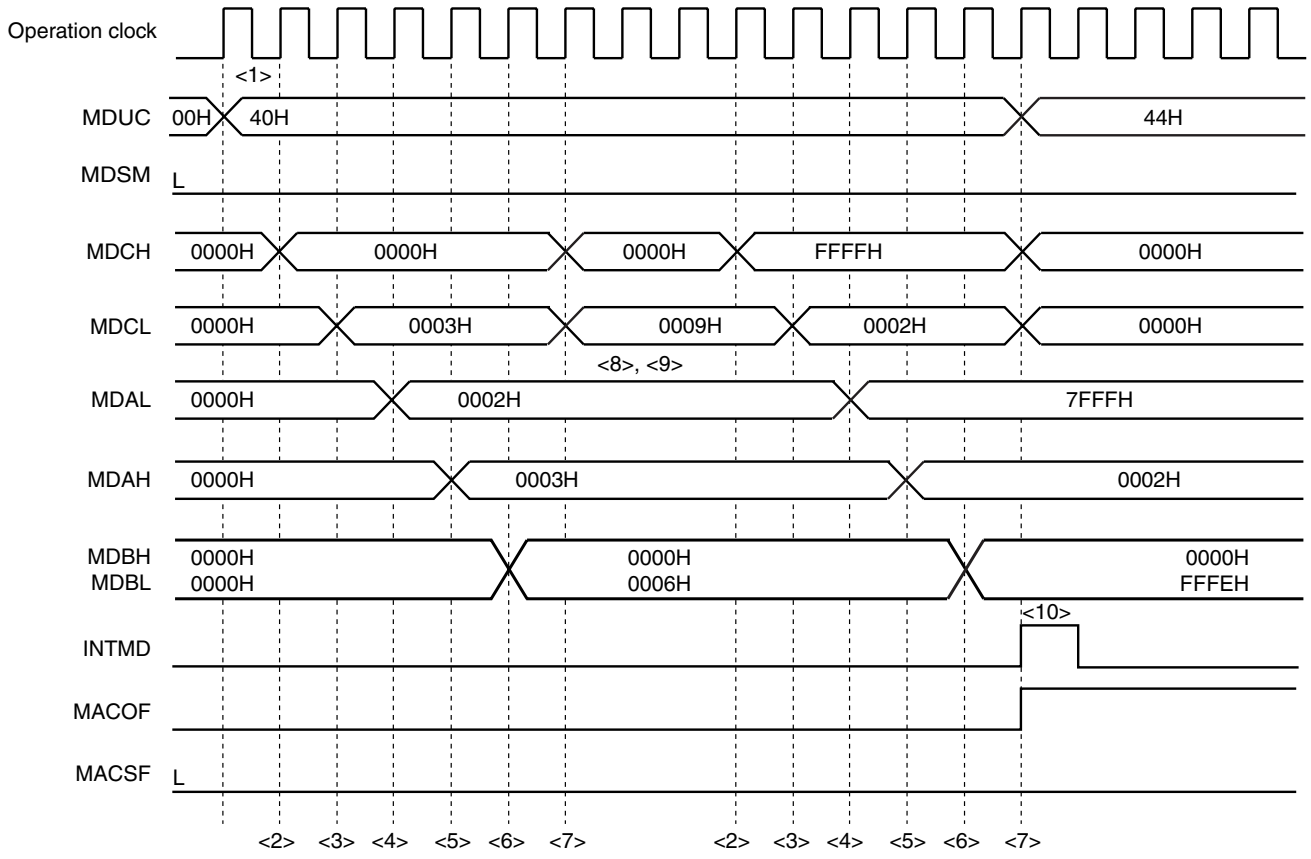


### 14.4.3 Multiply-accumulation (unsigned) operation

- Initial setting
  - <1> Set the multiplication/division control register (MDUC) to 40H.
  - <2> Set the initial accumulated value of higher 16 bits to multiplication/division data register C (L) (MDCL).
  - <3> Set the initial accumulated value of lower 16 bits to multiplication/division data register C (H) (MDCH).
  - <4> Set the multiplicand to multiplication/division data register A (L) (MDAL).
  - <5> Set the multiplier to multiplication/division data register A (H) (MDAH).
    (There is no preference in the order of executing steps <2>, <3>, and <4>. Multiplication operation is automatically started when the multiplier is set to the MDAH register, respectively.)
- During operation processing
  - <6> The multiplication operation finishes in one clock cycle.
    (The multiplication result is stored in multiplication/division data register B (L) (MDBL) and multiplication/division data register B (H) (MDBH).)
  - <7> After <6>, the multiply-accumulation operation finishes in one additional clock cycle. (There is a wait of at least two clock cycles after specifying the initial settings is finished (<5>).)
- Operation end
  - <8> Read the accumulated value (lower 16 bits) from the MDCL register.
  - <9> Read the accumulated value (higher 16 bits) from the MDCH register.
    (There is no preference in the order of executing steps <8> and <9>.)
  - (<10> If the result of the multiply-accumulation operation causes an overflow, the MACOF bit is set to 1, INTMD signal is occurred.)
- Next operation
  - <11> Start with the initial settings of each step to change the operation mode. When the same operation mode is used sequentially, settings <1> to <4> can be omitted.

**Remark** Steps <1> to <10> correspond to <1> to <10> in Figure 14-8.

**Figure 14-8. Timing Diagram of Multiply-Accumulation (Unsigned) Operation**
**(2 × 3 + 3 = 9 → 32767 × 2 + 4294901762 = 0 (over flow generated))**



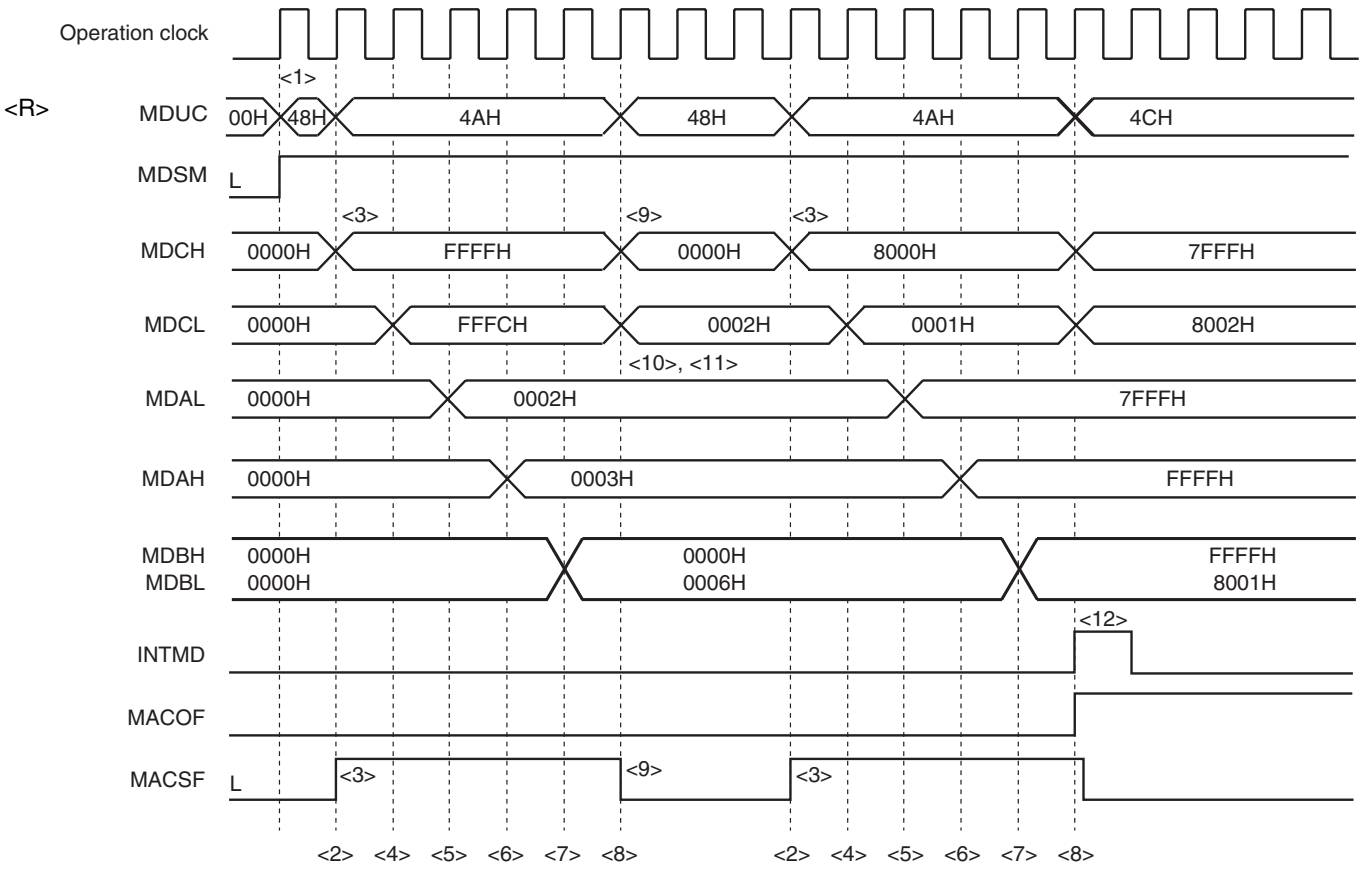
### 14.4.4 Multiply-accumulation (signed) operation

- Initial setting
  - <1> Set the multiplication/division control register (MDUC) to 48H.
  - <2> Set the initial accumulated value of higher 16 bits to multiplication/division data register C (H) (MDCH).
    (<3> If the accumulated value in the MDCH register is negative, the MACSF bit is set to 1.)
  - <4> Set the initial accumulated value of lower 16 bits to multiplication/division data register C (L) (MDCL).
  - <5> Set the multiplicand to multiplication/division data register A (L) (MDAL).
  - <6> Set the multiplier to multiplication/division data register A (H) (MDAH).
    (There is no preference in the order of executing steps <2>, <4>, and <5>. Multiplication operation is automatically started when the multiplier is set to the MDAH register of <6>, respectively.)
- During operation processing
  - <7> The multiplication operation finishes in one clock cycle.
    (The multiplication result is stored in multiplication/division data register B (L) (MDBL) and multiplication/division data register B (H) (MDBH).)
  - <8> After <7>, the multiply-accumulation operation finishes in one additional clock cycle. (There is a wait of at least two clock cycles after specifying the initial settings is finished (<6>).)
- Operation end
  - <9> If the accumulated value stored in the MDCL and MDCH registers is positive, the MACSF bit is cleared to 0.
  - <10> Read the accumulated value (lower 16 bits) from the MDCL register.
  - <11> Read the accumulated value (higher 16 bits) from the MDCH register.
    (There is no preference in the order of executing steps <10> and <11>.)
  - (<12> If the result of the multiply-accumulation operation causes an overflow, the MACOF bit is set to 1, INTMD signal is occurred.)
- Next operation
  - <13> Start with the initial settings of each step to change the operation mode. When the same operation mode is used sequentially, settings <1> to <5> can be omitted.

**Caution The data is in the two's complement format in multiply-accumulation (signed) operation.**

**Remark** Steps <1> to <12> correspond to <1> to <12> in Figure 14-9.

**Figure 14-9. Timing Diagram of Multiply-Accumulation (signed) Operation**
**(2 × 3 + (–4) = 2 → 32767 × (–1) + (–2147483647) = –2147450882 (overflow occurs.))**



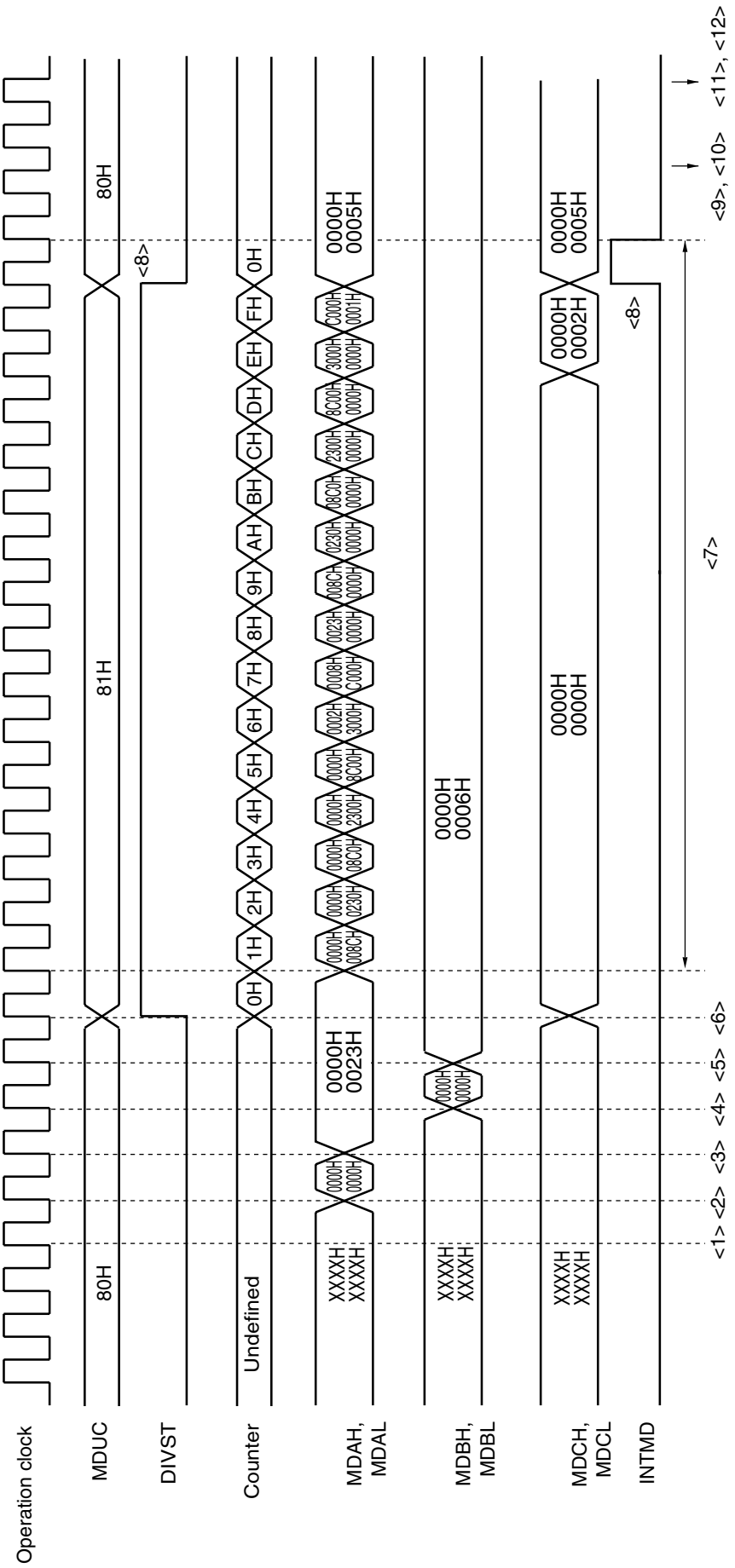
### 14.4.5 Division operation

- Initial setting
  - <1> Set the multiplication/division control register (MDUC) to 80H.
  - <2> Set the dividend (higher 16 bits) to multiplication/division data register A (H) (MDAH).
  - <3> Set the dividend (lower 16 bits) to multiplication/division data register A (L) (MDAL).
  - <4> Set the divisor (higher 16 bits) to multiplication/division data register B (H) (MDBH).
  - <5> Set the divisor (lower 16 bits) to multiplication/division data register B (L) (MDBL).
  - <6> Set bit 0 (DIVST) of the MDUC register to 1.
    (There is no preference in the order of executing steps <2> to <5>.)
- During operation processing
  - <7> The operation will end when one of the following processing is completed.
    - A wait of at least 16 clocks (The operation will end when 16 clocks have been issued.)
    - A check whether the DIVST bit has been cleared

    (The read values of the MDBL, MDBH, MDCL, and MDCH registers during operation processing are not guaranteed.)
- Operation end
  - <8> The DIVST bit is cleared and the operation ends. At this time, an interrupt request signal (INTMD) is generated if the operation was performed with MACMODE = 0.
  - <9> Read the quotient (lower 16 bits) from the MDAL register.
  - <10> Read the quotient (higher 16 bits) from the MDAH register.
  - <11> Read the remainder (lower 16 bits) from multiplication/division data register C (L) (MDCL).
  - <12> Read the remainder (higher 16 bits) from multiplication/division data register C (H) (MDCH).
    (There is no preference in the order of executing steps <9> to <12>.)
- Next operation
  - <13> Start with the initial settings of each step to change the operation mode. When the same operation mode is used sequentially, settings <1> to <5> can be omitted.

**Remark** Steps <1> to <12> correspond to <1> to <12> in Figure 14-10.

**Figure 14-10. Timing Diagram of Division Operation (Example: 35 ÷ 6 = 5, Remainder 5)**



# CHAPTER 15 DMA CONTROLLER

The RL78/G1A has an internal DMA (Direct Memory Access) controller.

Data can be automatically transferred between the peripheral hardware supporting DMA, SFRs, and internal RAM without via CPU.

As a result, the normal internal operation of the CPU and data transfer can be executed in parallel with transfer between the SFR and internal RAM, and therefore, a large capacity of data can be processed. In addition, real-time control using communication, timer, and A/D can also be realized.

## 15.1 Functions of DMA Controller

- Number of DMA channels: 2 channels
- Transfer unit: 8 or 16 bits
- Maximum transfer unit: 1024 times
- Transfer type: 2-cycle transfer (One transfer is processed in 2 clocks and the CPU stops during that processing.)
- Transfer mode: Single-transfer mode
- Transfer request: Selectable from the following peripheral hardware interrupts
  - A/D converter
  - Serial interface
    (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21, UART0 to UART2)
  - Timer (channel 0, 1, 2, 3)
- Transfer target: Between SFR and internal RAM

Here are examples of functions using DMA.

- Successive transfer of serial interface
- <R> Consecutive capturing of A/D conversion results
- Capturing port value at fixed interval

## 15.2 Configuration of DMA Controller

The DMA controller includes the following hardware.

**Table 15-1. Configuration of DMA Controller**

| Item | Configuration |
|---|---|
| Address registers | • DMA SFR address registers 0, 1 (DSA0, DSA1)<br>• DMA RAM address registers 0, 1 (DRA0, DRA1) |
| Count register | • DMA byte count registers 0, 1 (DBC0, DBC1) |
| Control registers | • DMA mode control registers 0, 1 (DMC0, DMC1)<br>• DMA operation control register 0, 1 (DRC0, DRC1) |

### 15.2.1 DMA SFR address register n (DSAn)

This is an 8-bit register that is used to set an SFR address that is the transfer source or destination of DMA channel n.

Set the lower 8 bits of the SFR addresses FFF00H to FFFFFH.

This register is not automatically incremented but fixed to a specific value.

In the 16-bit transfer mode, the least significant bit is ignored and is treated as an even address.

The DSAn register can be read or written in 8-bit units. However, it cannot be written during DMA transfer.

Reset signal generation clears this register to 00H.

**Figure 15-1. Format of DMA SFR Address Register n (DSAn)**

Address: FFFB0H (DSA0), FFFB1H (DSA1) After reset: 00H R/W

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DSAn | | | | | | | | |

**Remark** n: DMA channel number (n = 0, 1)

### 15.2.2 DMA RAM address register n (DRAn)

This is a 16-bit register that is used to set a RAM address that is the transfer source or destination of DMA channel n.

Addresses of the internal RAM area other than the general-purpose registers (see **Table 15-2**) can be set to this register.

Set the lower 16 bits of the RAM address.

This register is automatically incremented when DMA transfer has been started. It is incremented by +1 in the 8-bit transfer mode and by +2 in the 16-bit transfer mode. DMA transfer is started from the address set to this DRAn register. When the data of the last address has been transferred, the DRAn register stops with the value of the last address +1 in the 8-bit transfer mode, and the last address +2 in the 16-bit transfer mode.

In the 16-bit transfer mode, the least significant bit is ignored and is treated as an even address.

The DRAn register can be read or written in 8-bit or 16-bit units. However, it cannot be written during DMA transfer.

Reset signal generation clears this register to 0000H.

**Figure 15-2. Format of DMA RAM Address Register n (DRAn)**

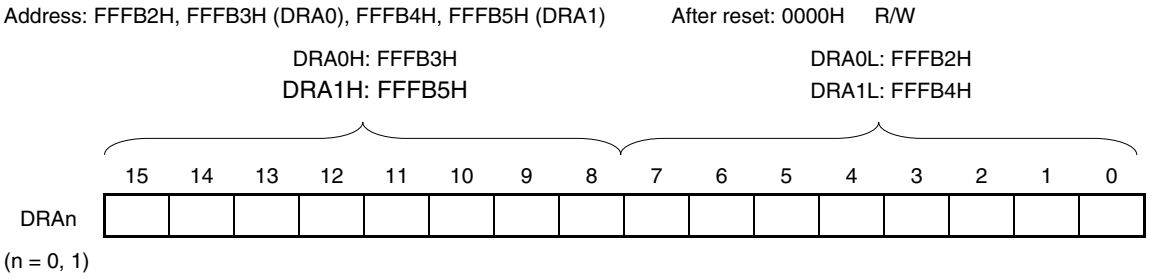



**Table 15-2. Internal RAM Area other than the General-purpose Registers**

| Part Number | Internal RAM Area other than the General-purpose Registers |
|---|---|
| R5F10ExA (x = 8, B, G)<br>R5F10ExC (x = 8, B, G, L) | FF700H to FFEDFH |
| R5F10ExD (x = 8, B, G, L) | FF300H to FFEDFH |
| R5F10ExE (x = 8, B, G, L) | FEF00H to FFEDFH |

**Remark** n: DMA channel number (n = 0, 1)

### 15.2.3 DMA byte count register n (DBCn)

This is a 10-bit register that is used to set the number of times DMA channel n executes transfer. Be sure to set the number of times of transfer to this DBCn register before executing DMA transfer (up to 1024 times).

Each time DMA transfer has been executed, this register is automatically decremented. By reading this DBCn register during DMA transfer, the remaining number of times of transfer can be learned.

The DBCn register can be read or written in 8-bit or 16-bit units. However, it cannot be written during DMA transfer.

Reset signal generation clears this register to 0000H.

**Figure 15-3. Format of DMA Byte Count Register n (DBCn)**

Address: FFFB6H, FFFB7H (DBC0), FFFB8H, FFFB9H (DBC1) After reset: 0000H R/W

DBC0H: FFFB7H
DBC1H: FFFB9H

DBC0L: FFFB6H
DBC1L: FFFB8H

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DBCn | 0 | 0 | 0 | 0 | 0 | 0 | | | | | | | | | | |

(n = 0, 1)

| DBCn[9:0] | Number of Times of Transfer (When DBCn is Written) | Remaining Number of Times of Transfer (When DBCn is Read) |
|---|---|---|
| 000H | 1024 | Completion of transfer or waiting for 1024 times of DMA transfer |
| 001H | 1 | Waiting for remaining one time of DMA transfer |
| 002H | 2 | Waiting for remaining two times of DMA transfer |
| 003H | 3 | Waiting for remaining three times of DMA transfer |
| • | • | • |
| • | • | • |
| • | • | • |
| 3FEH | 1022 | Waiting for remaining 1022 times of DMA transfer |
| 3FFH | 1023 | Waiting for remaining 1023 times of DMA transfer |

**Cautions 1. Be sure to clear bits 15 to 10 to "0".**

**2. If the general-purpose register is specified or the internal RAM space is exceeded as a result of continuous transfer, the general-purpose register or SFR space are written or read, resulting in loss of data in these spaces. Be sure to set the number of times of transfer that is within the internal RAM space.**

**Remark** n: DMA channel number (n = 0, 1)

## 15.3 Registers Controlling DMA Controller

DMA controller is controlled by the following registers.

- DMA mode control register n (DMCn)
- DMA operation control register n (DRCn)

**Remark** n: DMA channel number (n = 0, 1)

### 15.3.1 DMA mode control register n (DMCn)

The DMCn register is a register that is used to set a transfer mode of DMA channel n. It is used to select a transfer direction, data size, setting of pending, and start source. Bit 7 (STGn) is a software trigger that starts DMA.

Rewriting bits 6, 5, and 3 to 0 of the DMCn register is prohibited during operation (when DSTn = 1).

The DMCn register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 15-4. Format of DMA Mode Control Register n (DMCn) (1/2)**

Address: FFFBAH (DMC0), FFFBBH (DMC1) After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DMCn | STGn | DRSn | DSn | DWAITn | IFCn3 | IFCn2 | IFCn1 | IFCn0 |

| STGn[Note 1] | DMA transfer start software trigger |
|---|---|
| 0 | No trigger operation |
| 1 | DMA transfer is started when DMA operation is enabled (DENn = 1). |
| DMA transfer is performed once by writing 1 to the STGn bit when DMA operation is enabled (DENn = 1).<br>When this bit is read, 0 is always read. | |

| DRSn | Selection of DMA transfer direction |
|---|---|
| 0 | SFR to internal RAM |
| 1 | Internal RAM to SFR |

| DSn | Specification of transfer data size for DMA transfer |
|---|---|
| 0 | 8 bits |
| 1 | 16 bits |

| DWAITn[Note 2] | Pending of DMA transfer |
|---|---|
| 0 | Executes DMA transfer upon DMA start request (not held pending). |
| 1 | Holds DMA start request pending if any. |
| DMA transfer that has been held pending can be started by clearing the value of the DWAITn bit to 0.<br>It takes 2 clocks to actually hold DMA transfer pending when the value of the DWAITn bit is set to 1. | |

**Notes 1.** The software trigger (STGn) can be used regardless of the IFCn0 to IFCn3 bits values.

**2.** When DMA transfer is held pending while using two or more DMA channels, be sure to hold the DMA transfer pending for all channels (by setting the DWAIT0 and DWAIT1 bits to 1).

**Remark** n: DMA channel number (n = 0, 1)

**Figure 15-4. Format of DMA Mode Control Register n (DMCn) (2/2)**

Address: FFFBAH (DMC0), FFFBBH (DMC1) After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| DMCn | STGn | DRSn | DSn | DWAITn | IFCn3 | IFCn2 | IFCn1 | IFCn0 |

(When n = 0 or 1)

| IFCn 3 | IFCn 2 | IFCn 1 | IFCn 0 | Selection of DMA start source[Note] | |
|---|---|---|---|---|---|
| | | | | Trigger signal | Trigger contents |
| 0 | 0 | 0 | 0 | – | Disables DMA transfer by interrupt. (Only software trigger is enabled.) |
| 0 | 0 | 0 | 1 | INTAD | A/D conversion end interrupt |
| 0 | 0 | 1 | 0 | INTTM00 | End of timer channel 0 count or capture end interrupt |
| 0 | 0 | 1 | 1 | INTTM01 | End of timer channel 1 count or capture end interrupt |
| 0 | 1 | 0 | 0 | INTTM02 | End of timer channel 2 count or capture end interrupt |
| 0 | 1 | 0 | 1 | INTTM03 | End of timer channel 3 count or capture end interrupt |
| 0 | 1 | 1 | 0 | INTST0/INTCSI00 | UART0 transmission transfer end or buffer empty interrupt/CSI00 transfer end or buffer empty interrupt |
| 0 | 1 | 1 | 1 | INTSR0/INTCSI01 | UART0 reception transfer end interrupt/CSI01 transfer end or buffer empty interrupt |
| 1 | 0 | 0 | 0 | INTST1/INTCSI10 | UART1 transmission transfer end or buffer empty interrupt/CSI10 transfer end or buffer empty interrupt |
| 1 | 0 | 0 | 1 | INTSR1/INTCSI11 | UART1 reception transfer end interrupt/CSI11 transfer end or buffer empty interrupt |
| 1 | 0 | 1 | 0 | INTST2/INTCSI20 | UART2 transmission transfer end or buffer empty interrupt/CSI20 transfer end or buffer empty interrupt |
| 1 | 0 | 1 | 1 | INTSR2/INTCSI21 | UART2 reception transfer end interrupt/CSI21 transfer end or buffer empty interrupt |
| Other than above | | | | Setting prohibited | |

**Note** The software trigger (STGn) can be used regardless of the IFCn0 to IFCn3 bits values.

**Remark** n: DMA channel number (n = 0, 1)

### 15.3.2 DMA operation control register n (DRCn)

The DRCn register is a register that is used to enable or disable transfer of DMA channel n.

Rewriting bit 7 (DENn) of this register is prohibited during operation (when DSTn = 1).

The DRCn register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 15-5. Format of DMA Operation Control Register n (DRCn)**

Address: FFFBCH (DRC0), FFFBDH (DRC1) After reset: 00H R/W

| Symbol | <7> | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| DRCn | DENn | 0 | 0 | 0 | 0 | 0 | 0 | DSTn |

| DENn | DMA operation enable flag |
|---|---|
| 0 | Disables operation of DMA channel n (stops operating cock of DMA). |
| 1 | Enables operation of DMA channel n. |
| DMAC waits for a DMA trigger when DSTn = 1 after DMA operation is enabled (DENn = 1). | |

| DSTn | DMA transfer mode flag |
|---|---|
| 0 | DMA transfer of DMA channel n is completed. |
| 1 | DMA transfer of DMA channel n is not completed (still under execution). |
| DMAC waits for a DMA trigger when DSTn = 1 after DMA operation is enabled (DENn = 1).<br>When a software trigger (STGn) or the start source trigger set by the IFCn3 to IFCn0 bits is input, DMA transfer is started.<br>When DMA transfer is completed after that, this bit is automatically cleared to 0.<br>Write 0 to this bit to forcibly terminate DMA transfer under execution. | |

**Caution The DSTn flag is automatically cleared to 0 when a DMA transfer is completed.**
**Writing the DENn flag is enabled only when DSTn = 0. When a DMA transfer is terminated without waiting for generation of the interrupt (INTDMAn) of DMAn, therefore, set the DSTn bit to 0 and then the DENn bit to 0 (for details, see 15.5.5 Forced termination by software).**

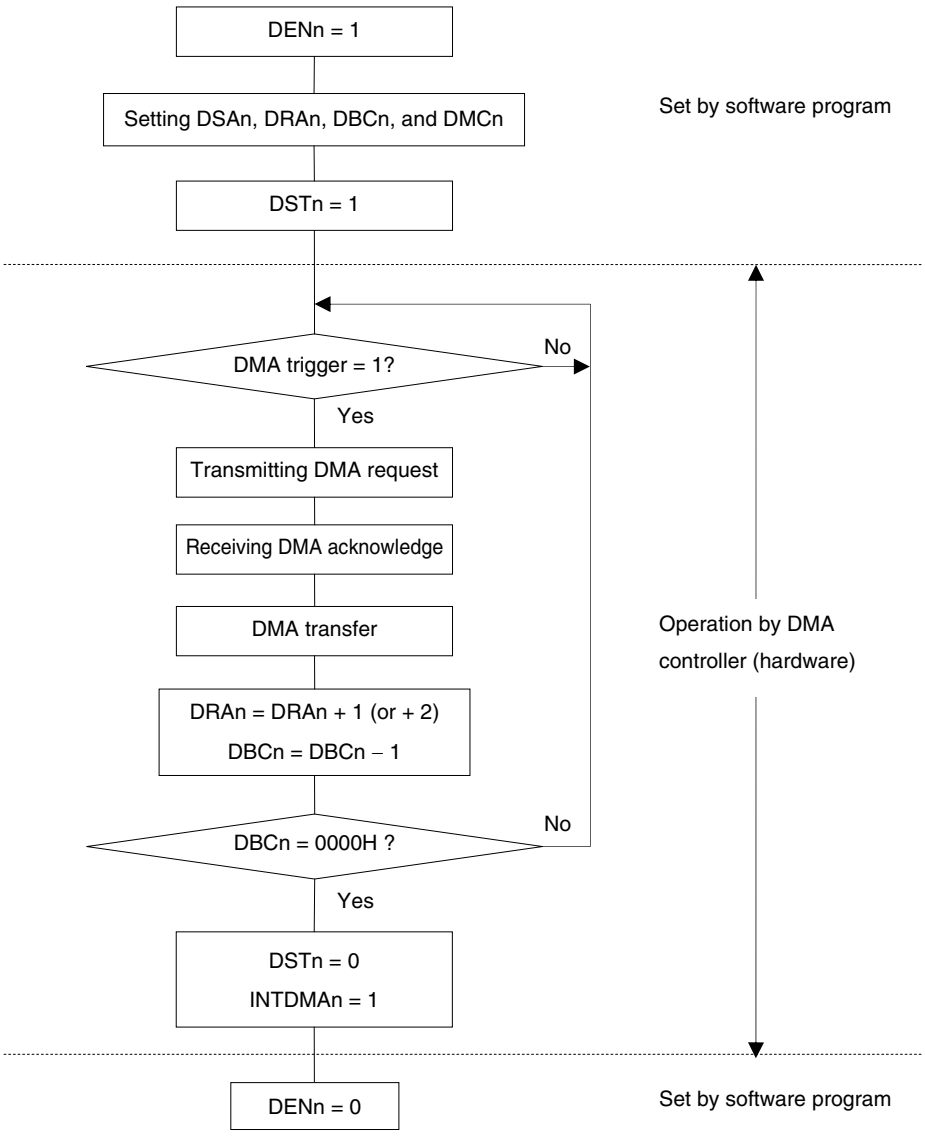
**Remark** n: DMA channel number (n = 0, 1)

## 15.4 Operation of DMA Controller

### 15.4.1 Operation procedure

<1> The DMA controller is enabled to operate when DENn = 1. Before writing the other registers, be sure to set the DENn bit to 1. Use 80H to write with an 8-bit manipulation instruction.
<2> Set an SFR address, a RAM address, the number of times of transfer, and a transfer mode of DMA transfer to DMA SFR address register n (DSAn), DMA RAM address register n (DRAn), DMA byte count register n (DBCn), and DMA mode control register n (DMCn).
<3> The DMA controller waits for a DMA trigger when DSTn = 1. Use 81H to write with an 8-bit manipulation instruction.
<4> When a software trigger (STGn) or a start source trigger specified by the IFCn3 to IFCn0 bits is input, a DMA transfer is started.
<5> Transfer is completed when the number of times of transfer set by the DBCn register reaches 0, and transfer is automatically terminated by occurrence of an interrupt (INTDMAn).
<6> Stop the operation of the DMA controller by clearing the DENn bit to 0 when the DMA controller is not used.

**Figure 15-6. Operation Procedure**




**Remark** n: DMA channel number (n = 0, 1)

### 15.4.2 Transfer mode

The following four modes can be selected for DMA transfer by using bits 6 and 5 (DRSn and DSn) of DMA mode control register n (DMCn).

| DRSn | DSn | DMA Transfer Mode |
|---|---|---|
| 0 | 0 | Transfer from SFR of 1-byte data (fixed address) to RAM (address is incremented by +1) |
| 0 | 1 | Transfer from SFR of 2-byte data (fixed address) to RAM (address is incremented by +2) |
| 1 | 0 | Transfer from RAM of 1-byte data (address is incremented by +1) to SFR (fixed address) |
| 1 | 1 | Transfer from RAM of 2-byte data (address is incremented by +2) to SFR (fixed address) |

By using these transfer modes, up to 1024 bytes of data can be consecutively transferred by using the serial interface, data resulting from A/D conversion can be consecutively transferred, and port data can be scanned at fixed time intervals by using a timer.

### 15.4.3 Termination of DMA transfer

When DBCn = 00H and DMA transfer is completed, the DSTn bit is automatically cleared to 0. An interrupt request (INTDMAn) is generated and transfer is terminated.

When the DSTn bit is cleared to 0 to forcibly terminate DMA transfer, DMA byte count register n (DBCn) and DMA RAM address register n (DRAn) hold the value when transfer is terminated.

The interrupt request (INTDMAn) is not generated if transfer is forcibly terminated.

**Remark** n: DMA channel number (n = 0, 1)

## 15.5 Example of Setting of DMA Controller

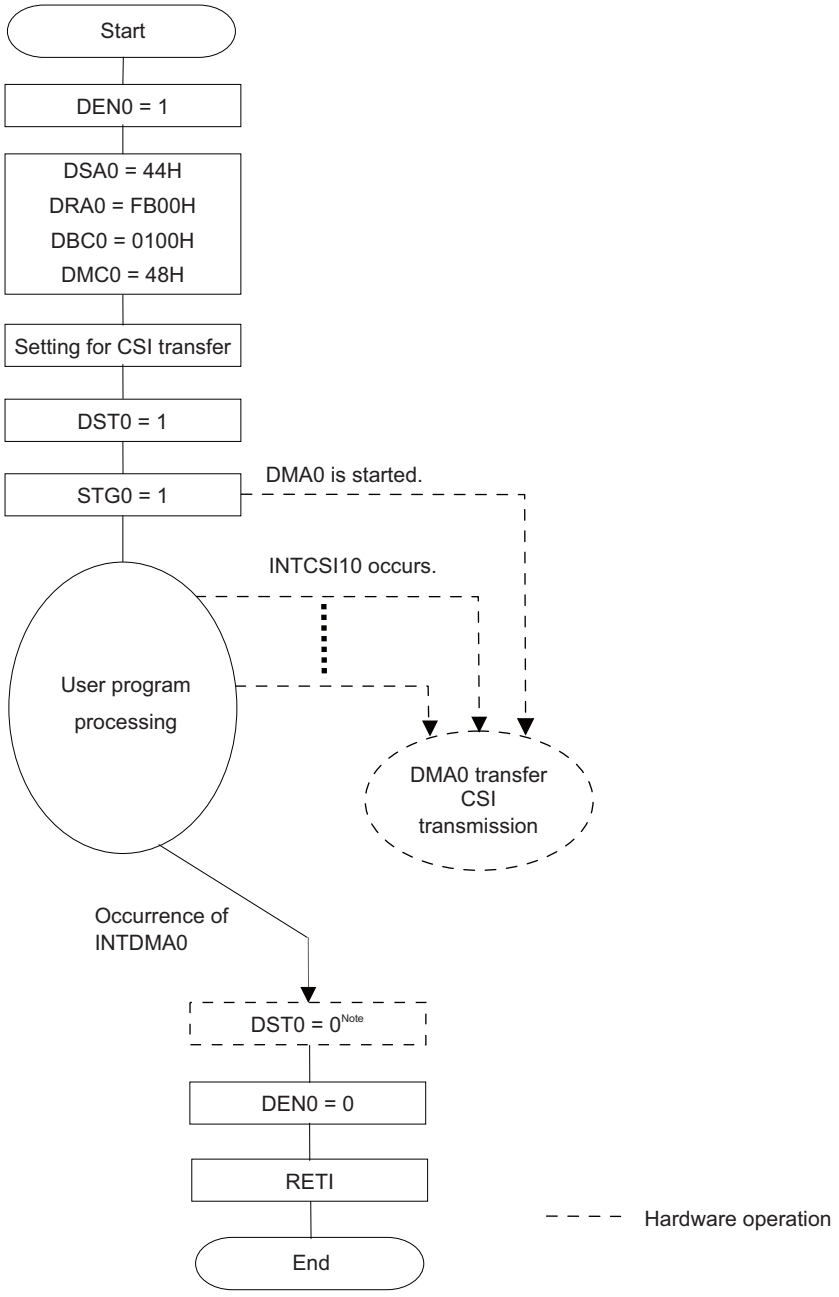
### 15.5.1 CSI consecutive transmission

A flowchart showing an example of setting for CSI consecutive transmission is shown below.

- Consecutive transmission of CSI10 (256 bytes)
- DMA channel 0 is used for DMA transfer.
- DMA start source: INTCSI10 (software trigger (STG0) only for the first start source)
- Interrupt of CSI10 is specified by IFC03 to IFC00 = 1000B.
- Transfers FFB00H to FFBFFH (256 bytes) of RAM to FFF44H of the data register (SIO10) of CSI.

**Remark** IFC03 to IFC00: Bits 3 to 0 of DMA mode control registers 0 (DMC0)

**Figure 15-7. Example of Setting for CSI Consecutive Transmission**




**Note** The DST0 flag is automatically cleared to 0 when a DMA transfer is completed.
Writing the DEN0 flag is enabled only when DST0 = 0. To terminate a DMA transfer without waiting for occurrence of the interrupt of DMA0 (INTDMA0), set the DST0 bit to 0 and then the DEN0 bit to 0 (for details, see **15.5.5 Forced termination by software**).

The fist trigger for consecutive transmission is not started by the interrupt of CSI. In this example, it start by a software trigger.

CSI transmission of the second time and onward is automatically executed.

A DMA interrupt (INTDMA0) occurs when the last transmit data has been written to the data register.

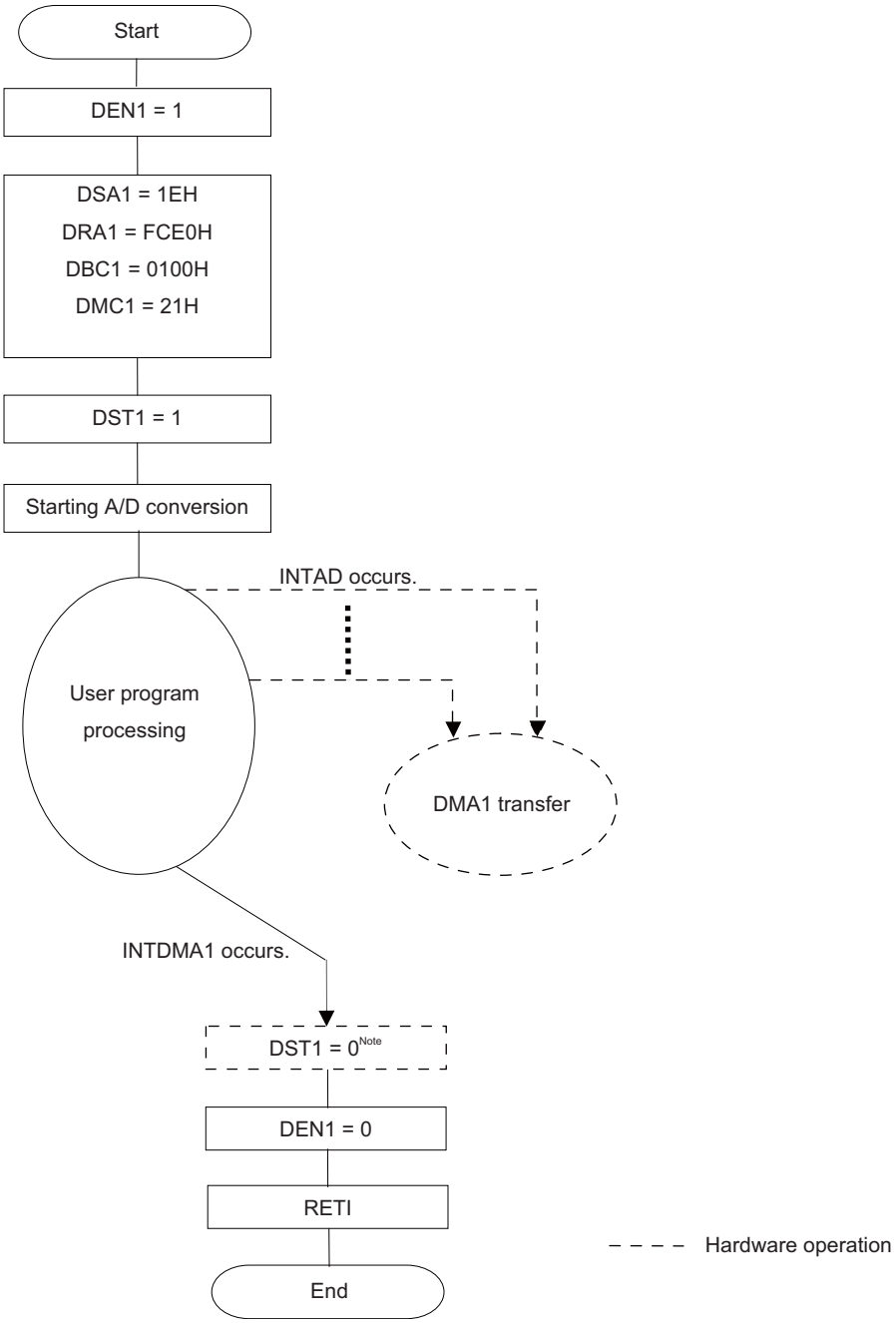
### 15.5.2 Consecutive capturing of A/D conversion results

A flowchart of an example of setting for consecutively capturing A/D conversion results is shown below.

- Consecutive capturing of A/D conversion results.
- DMA channel 1 is used for DMA transfer.
- DMA start source: INTAD
- Interrupt of A/D is specified by IFC13 to IFC10 = 0001B.
- Transfers FFF1EH and FFF1FH (2 bytes) of the 12-bit A/D conversion result register (ADCR) to 512 bytes of FFCE0H to FFEDFH of RAM.

**Remark** IFC13 to IFC10: Bits 3 to 0 of DMA mode control registers 1 (DMC1)

**Figure 15-8. Example of Setting of Consecutively Capturing A/D Conversion Results**



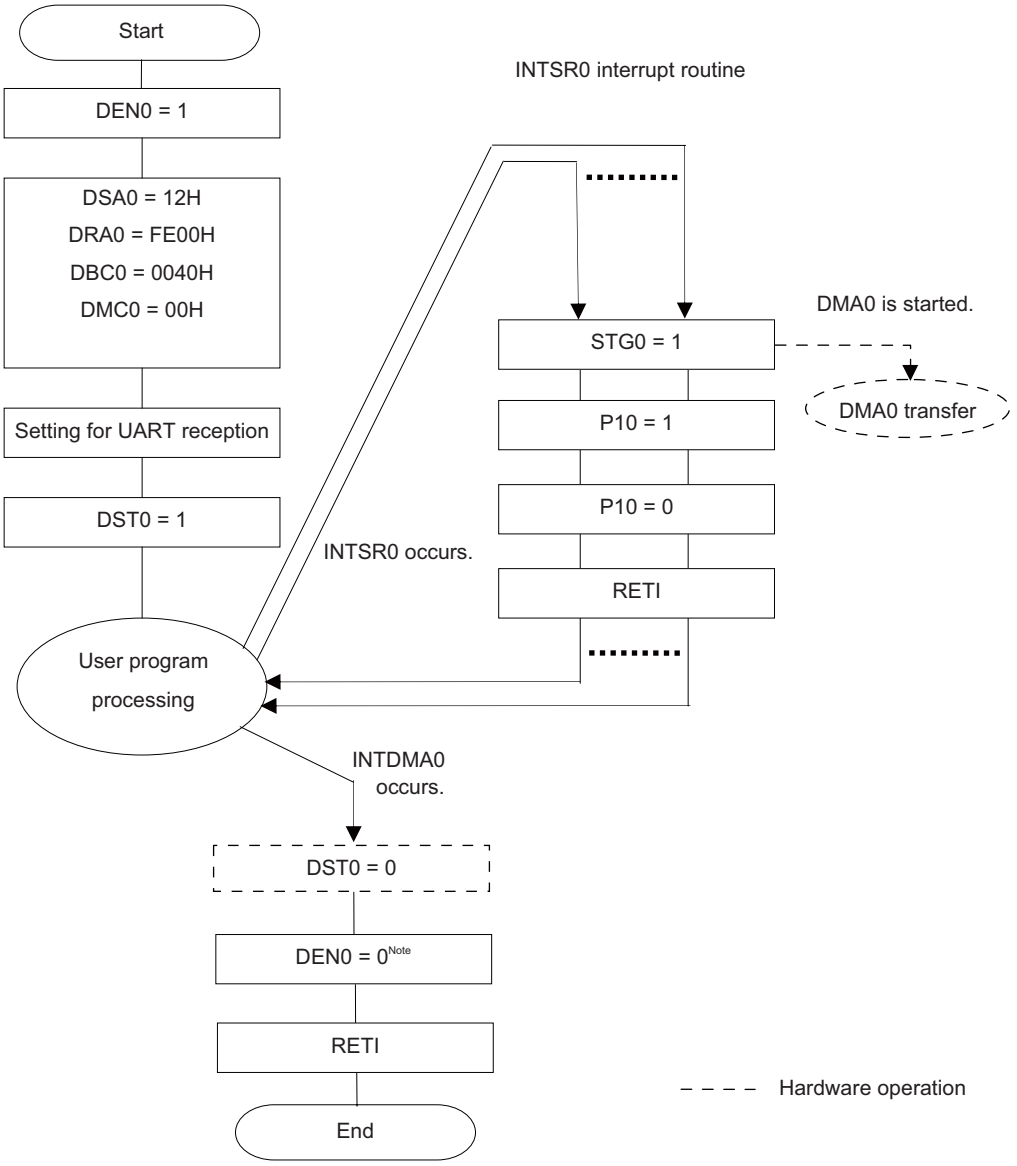

**Note** The DST1 flag is automatically cleared to 0 when a DMA transfer is completed.
Writing the DEN1 flag is enabled only when DST1 = 0. To terminate a DMA transfer without waiting for occurrence of the interrupt of DMA1 (INTDMA1), set the DST1 bit to 0 and then the DEN1 bit to 0 (for details, see **15.5.5 Forced termination by software**).

### 15.5.3 UART consecutive reception + ACK transmission

A flowchart illustrating an example of setting for UART consecutive reception + ACK transmission is shown below.

- Consecutively receives data from UART0 and outputs ACK to P10 on completion of reception.
- DMA channel 0 is used for DMA transfer.
- DMA start source: Software trigger (DMA transfer on occurrence of an interrupt is disabled.)
- Transfers FFF12H of UART receive data register 0 (RXD0) to 64 bytes of FFE00H to FFE3FH of RAM.

<R> **Figure 15-9. Example of Setting for UART Consecutive Reception + ACK Transmission**




**Note** The DST0 flag is automatically cleared to 0 when a DMA transfer is completed.
Writing the DEN0 flag is enabled only when DST0 = 0. To terminate a DMA transfer without waiting for occurrence of the interrupt of DMA0 (INTDMA0), set the DST0 bit to 0 and then the DEN0 bit to 0 (for details, see **15.5.5 Forced termination by software**).

**Remark** This is an example where a software trigger is used as a DMA start source.
If ACK is not transmitted and if only data is consecutively received from UART, the UART reception end interrupt (INTSR0) can be used to start DMA for data reception.

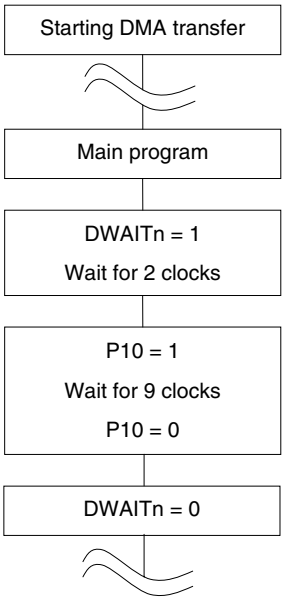
### 15.5.4 Holding DMA transfer pending by DWAITn bit

When DMA transfer is started, transfer is performed while an instruction is executed. At this time, the operation of the CPU is stopped and delayed for the duration of 2 clocks. If this poses a problem to the operation of the set system, a DMA transfer can be held pending by setting the DWAITn bit to 1. The DMA transfer for a transfer trigger that occurred while DMA transfer was held pending is executed after the pending status is canceled. However, because only one transfer trigger can be held pending for each channel, even if multiple transfer triggers occur for one channel during the pending status, only one DMA transfer is executed after the pending status is canceled.

To output a pulse with a width of 10 clocks of the operating frequency from the P10 pin, for example, the clock width increases to 12 if a DMA transfer is started midway. In this case, the DMA transfer can be held pending by setting the DWAITn bit to 1.

After setting the DWAITn bit to 1, it takes two clocks until a DMA transfer is held pending.

**Figure 15-10. Example of Setting for Holding DMA Transfer Pending by DWAITn Bit**




**Caution When DMA transfer is held pending while using two or more DMA channels, be sure to held the DMA transfer pending for all channels (by setting DWAIT0 and DWAIT1 to 1). If the DMA transfer of one channel is executed while that of the other channel is held pending, DMA transfer might not be held pending for the latter channel.**

**Remarks 1.** n: DMA channel number (n = 0, 1)

**2.** 1 clock: $1/f_{CLK}$ ($f_{CLK}$: CPU clock)

### 15.5.5 Forced termination by software

After the DSTn bit is set to 0 by software, it takes up to 2 clocks until a DMA transfer is actually stopped and the DSTn bit is set to 0. To forcibly terminate a DMA transfer by software without waiting for occurrence of the interrupt (INTDMAn) of DMAn, therefore, perform either of the following processes.

<When using one DMA channel>

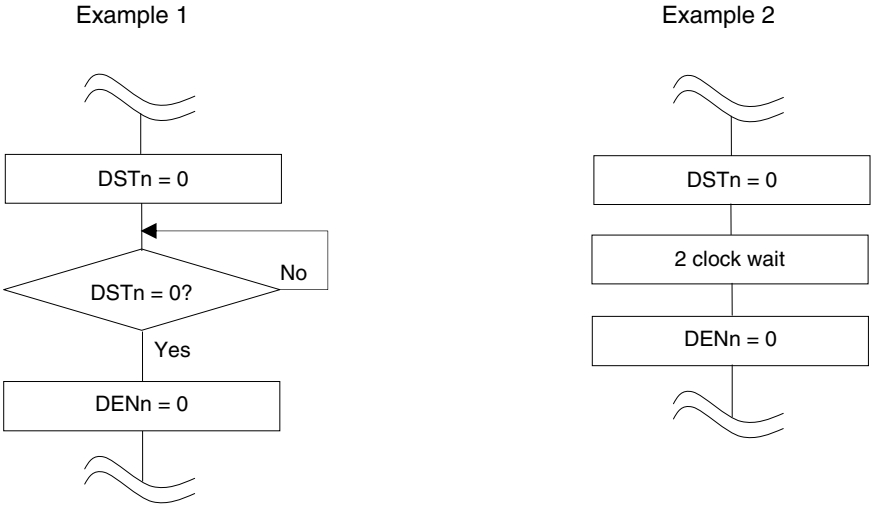
- Set the DSTn bit to 0 (use DRCn = 80H to write with an 8-bit manipulation instruction) by software, confirm by polling that the DSTn bit has actually been cleared to 0, and then set the DENn bit to 0 (use DRCn = 00H to write with an 8-bit manipulation instruction).

- Set the DSTn bit to 0 (use DRCn = 80H to write with an 8-bit manipulation instruction) by software and then set the DENn bit to 0 (use DRCn = 00H to write with an 8-bit manipulation instruction) two or more clocks after.

<When using two or more DMA channels>

- To forcibly terminate DMA transfer by software when using two or more DMA channels (by setting DSTn to 0), clear the DSTn bit to 0 after the DMA transfer is held pending by setting the DWAITn bits of all using channels to 1. Next, clear the DWAITn bits of all using channels to 0 to cancel the pending status, and then clear the DENn bit to 0.

**Figure 15-11. Forced Termination of DMA Transfer (1/2)**



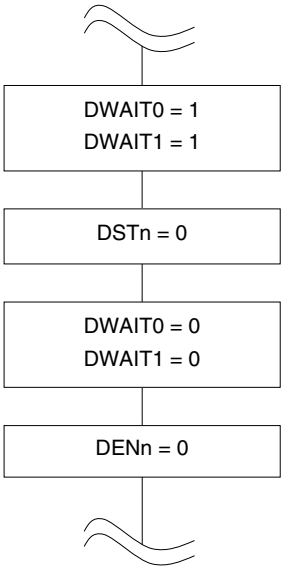

**Remarks 1.** n: DMA channel number (n = 0, 1)

**2.** 1 clock: 1/$f_{CLK}$ ($f_{CLK}$: CPU clock)

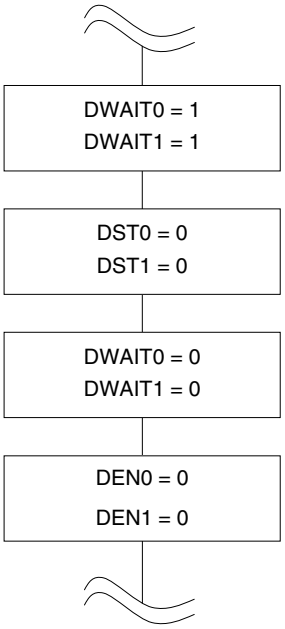
**Figure 15-11. Forced Termination of DMA Transfer (2/2)**

Example 3

- Procedure for forcibly terminating the DMA transfer for one channel if both channels are used




- Procedure for forcibly terminating the DMA transfer for both channels if both channels are used




**Caution In example 3, the system is not required to wait two clock cycles after the DWAITn bit is set to 1. In addition, the system does not have to wait two clock cycles after clearing the DSTn bit to 0, because more than two clock cycles elapse from when the DSTn bit is cleared to 0 to when the DENn bit is cleared to 0.**

**Remarks 1.** n: DMA channel number (n = 0, 1)

**2.** 1 clock: $1/f_{CLK}$ ($f_{CLK}$: CPU clock)

## 15.6 Cautions on Using DMA Controller

### (1) Priority of DMA

During DMA transfer, a request from the other DMA channel is held pending even if generated. The pending DMA transfer is started after the ongoing DMA transfer is completed. If two or more DMA requests are generated at the same time, however, their priority are DMA channel 0 > DMA channel 1.

If a DMA request and an interrupt request are generated at the same time, the DMA transfer takes precedence, and then interrupt servicing is executed.

### (2) DMA response time

The response time of DMA transfer is as follows.

**Table 15-3. Response Time of DMA Transfer**

| | Minimum Time | Maximum Time |
|---|---|---|
| Response time | 3 clocks | 10 clocks[Note] |

**Note** The maximum time necessary to execute an instruction from internal RAM is 16 clock cycles.

**Cautions 1. The above response time does not include the two clock cycles required for a DMA transfer.**

**2. When executing a DMA pending instruction (see 15.6 (4)), the maximum response time is extended by the execution time of that instruction to be held pending.**

**3. Do not specify successive transfer triggers for a channel within a period equal to the maximum response time plus one clock cycle, because they might be ignored.**

**Remark** 1 clock: $1/f_{CLK}$ ($f_{CLK}$: CPU clock)

### (3) Operation in standby mode

The DMA controller operates as follows in the standby mode.

**Table 15-4. DMA Operation in Standby Mode**

| Status | DMA Operation |
|---|---|
| HALT mode | Normal operation |
| STOP mode | Stops operation.<br>If DMA transfer and STOP instruction execution contend, DMA transfer may be damaged. Therefore, stop DMA before executing the STOP instruction. |

**(4) DMA pending instruction**

Even if a DMA request is generated, DMA transfer is held pending immediately after the following instructions.

<R>
- CALL !addr16
- CALL $!addr20
- CALL !!addr20
- CALL rp
- CALLT [addr5]
- BRK
- MOV PSW, #byte
- MOV PSW, A
- MOV1 PSW. bit, CY
- SET1 PSW. bit
- CLR1 PSW. bit
- POP PSW
- BTCLR PSW. bit, $addr20
- EI
- DI
- Write instructions for registers IF0L, IF0H, IF1L, IF1H, IF2L, IF2H, MK0L, MK0H, MK1L, MK1H, MK2L, MK2H, PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR10L, PR10H, PR11L, PR11H, PR12L, and PR12H each.
- Instruction for accessing the data flash memory

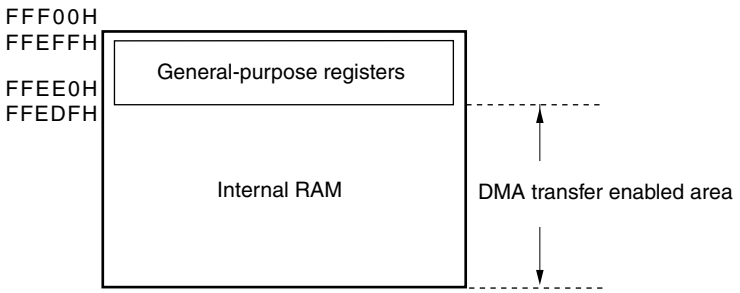
**(5) Operation if address in general-purpose register area or other than those of internal RAM area is specified**

The address indicated by DMA RAM address register n (DRAn) is incremented during DMA transfer. If the address is incremented to an address in the general-purpose register area or exceeds the area of the internal RAM, the following operation is performed.

- In mode of transfer from SFR to RAM
  The data of that address is lost.

- In mode of transfer from RAM to SFR
  Undefined data is transferred to SFR.

In either case, malfunctioning may occur or damage may be done to the system. Therefore, make sure that the address is within the internal RAM area other than the general-purpose register area.



**(6) Operation if instructions for accessing the data flash area**

<R> If the data flash area is accessed after a next instruction execution from start of DMA transfer, a 3-clock wait will be inserted to the next instruction.

```
Instruction 1
DMA transfer
Instruction 2   ◄─The wait of three clock cycles occurs.
MOV A,          ! DataFlash area
```

# CHAPTER 16 INTERRUPT FUNCTIONS

The interrupt function switches the program execution to other processing. When the branch processing is finished, the program returns to the interrupted processing.

The number of interrupt sources differs, depending on the product.

| | | 25-pin | 32-pin | 48-pin | 64-pin |
|---|---|---|---|---|---|
| Maskable interrupts | External | 5 | 6 | 10 | 13 |
| | Internal | 24 | 27 | 27 | 27 |

## 16.1 Interrupt Function Types

The following two types of interrupt functions are used.

**(1) Maskable interrupts**

These interrupts undergo mask control. Maskable interrupts can be divided into four priority groups by setting the priority specification flag registers (PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR10L, PR10H, PR11L, PR11H, PR12L, PR12H).

Multiple interrupt servicing can be applied to low-priority interrupts when high-priority interrupts are generated. If two or more interrupt requests, each having the same priority, are simultaneously generated, then they are processed according to the default priority of vectored interrupt servicing. Default priority, see **Table 16-1**.

A standby release signal is generated and STOP, HALT, and SNOOZE modes are released.

External interrupt requests and internal interrupt requests are provided as maskable interrupts.

**(2) Software interrupt**

This is a vectored interrupt generated by executing the BRK instruction. It is acknowledged even when interrupts are disabled. The software interrupt does not undergo interrupt priority control.

## 16.2 Interrupt Sources and Configuration

Interrupt sources include maskable interrupts and software interrupts. In addition, they also have up to seven reset sources (see **Table 16-1**). The vector codes that store the program start address when branching due to the generation of a reset or various interrupt requests are two bytes each, so interrupts jump to a 64 K address of 00000H to 0FFFFH.

**Table 16-1. Interrupt Source List (1/3)**

| Interrupt Type | Default Priority[Note 1] | Interrupt Source | | Internal/ External | Vector Table Address | Basic Configuration Type[Note 2] | 64-pin | 48-pin | 32-pin | 25-pin |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Name | Trigger | | | | | | | |
| Maskable | 0 | INTWDTI | Watchdog timer interval[Note 3] (75% of overflow time+1/2$f_{IL}$) | Internal | 0004H | (A) | √ | √ | √ | √ |
| | 1 | INTLVI | Voltage detection[Note 4] | | 0006H | | √ | √ | √ | √ |
| | 2 | INTP0 | Pin input edge detection | External | 0008H | (B) | √ | √ | √ | √ |
| | 3 | INTP1 | | | 000AH | | √ | √ | √ | √ |
| | 4 | INTP2 | | | 000CH | | √ | √ | √ | √ |
| | 5 | INTP3 | | | 000EH | | √ | √ | √ | √ |
| | 6 | INTP4 | | | 0010H | | √ | √ | √ | √ |
| | 7 | INTP5 | | | 0012H | | √ | √ | – | – |
| | 8 | INTST2/ INTCSI20/ INTIIC20 | UART2 transmission transfer end or buffer empty interrupt/CSI20 transfer end or buffer empty interrupt/IIC20 transfer end | Internal | 0014H | (A) | √ | √ | √ | – |
| | 9 | INTSR2/ INTCSI21/ INTIIC21 | UART2 reception transfer end/CSI21 transfer end or buffer empty interrupt/IIC21 transfer end | | 0016H | | √ | √ | √[Note 5] | – |
| | 10 | INTSRE2 | UART2 reception communication error occurrence | | 0018H | | √ | √ | √ | – |
| | 11 | INTDMA0 | End of DMA0 transfer | | 001AH | | √ | √ | √ | √ |
| | 12 | INTDMA1 | End of DMA1 transfer | | 001CH | | √ | √ | √ | √ |
| | 13 | INTST0/ INTCSI00/ INTIIC00 | UART0 transmission transfer end or buffer empty interrupt/CSI00 transfer end or buffer empty interrupt/IIC00 transfer end | | 001EH | | √ | √ | √ | √ |
| | 14 | INTSR0/ INTCSI01/ INTIIC01 | UART0 reception transfer end/CSI01 transfer end or buffer empty interrupt/IIC01 transfer end | | 0020H | | √ | √ | √[Note 6] | √[Note 6] |
| | 15 | INTSRE0 | UART0 reception communication error occurrence | | 0022H | | √ | √ | √ | √ |
| | | INTTM01H | End of timer channel 1 count or capture (at higher 8-bit timer operation) | | | | √ | √ | √ | √ |

**Notes 1.** The default priority determines the sequence of interrupts if two or more maskable interrupts occur simultaneously. Zero indicates the highest priority and 39 indicates the lowest priority.

**2.** Basic configuration types (A) to (D) correspond to (A) to (D) in Figure 16-1.

**3.** When bit 7 (WDTINT) of the option byte (000C0H) is set to 1.

**4.** When bit 7 (LVIMD) of the voltage detection level register (LVIS) is cleared to 0.

**5.** INTSR2 only.

**6.** INTSR0 only.

**Table 16-1. Interrupt Source List (2/3)**

| Interrupt Type | Default Priority[Note 1] | Interrupt Source | | Internal/ External | Vector Table Address | Basic Configuration Type[Note 2] | 64-pin | 48-pin | 32-pin | 25-pin |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Name | Trigger | | | | | | | |
| Maskable | 16 | INTST1/ INTCSI10/ INTIIC10 | UART1 transmission transfer end or buffer empty interrupt/CSI10 transfer end or buffer empty interrupt/IIC10 transfer end | Internal | 0024H | (A) | √ | √[Note 3] | √[Note 3] | √[Note 3] |
| | 17 | INTSR1/ INTCSI11/ INTIIC11 | UART1 reception transfer end/CSI11 transfer end or buffer empty interrupt/IIC11 transfer end | | 0026H | | √ | √ | √ | √ |
| | 18 | INTSRE1 | UART1 reception communication error occurrence | | 0028H | | √ | √ | √ | √ |
| | | INTTM03H | End of timer channel 3 count or capture (at higher 8-bit timer operation) | | | | √ | √ | √ | √ |
| | 19 | INTIICA0 | End of IICA0 communication | | 002AH | | √ | √ | √ | √ |
| | 20 | INTTM00 | End of timer channel 0 count or capture | | 002CH | | √ | √ | √ | √ |
| | 21 | INTTM01 | End of timer channel 1 count or capture (at 16-bit/lower 8-bit timer operation) | | 002EH | | √ | √ | √ | √ |
| | 22 | INTTM02 | End of timer channel 2 count or capture | | 0030H | | √ | √ | √ | √ |
| | 23 | INTTM03 | End of timer channel 3 count or capture (at 16-bit/lower 8-bit timer operation) | | 0032H | | √ | √ | √ | √ |
| | 24 | INTAD | End of A/D conversion | | 0034H | | √ | √ | √ | √ |
| | 25 | INTRTC | Fixed-cycle signal of real-time clock/alarm match detection | | 0036H | | √ | √ | √ | √ |
| | 26 | INTIT | Interval signal of 12-bit interval timer detection | | 0038H | | √ | √ | √ | √ |
| | 27 | INTKR | Key return signal detection | External | 003AH | (C) | √ | √ | √ | (√)[Note 4] |
| | 28 | INTTM04 | End of timer channel 4 count or capture | Internal | 0042H | (A) | √ | √ | √ | √ |

**Notes 1.** The default priority determines the sequence of interrupts if two or more maskable interrupts occur simultaneously. Zero indicates the highest priority and 39 indicates the lowest priority.

**2.** Basic configuration types (A) to (D) correspond to (A) to (D) in Figure 16-1.

**3.** INTST1 only.

**4.** When setting peripheral I/O redirection register (PIOR)

**Table 16-1. Interrupt Source List (3/3)**

| Interrupt Type | Default Priority[Note 1] | Interrupt Source | | Internal/ External | Vector Table Address | Basic Configuration Type[Note 2] | 64-pin | 48-pin | 32-pin | 25-pin |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Name | Trigger | | | | | | | |
| Maskable | 29 | INTTM05 | End of timer channel 5 count or capture | Internal | 0044H | (A) | √ | √ | √ | √ |
| | 30 | INTTM06 | End of timer channel 6 count or capture | | 0046H | | √ | √ | √ | √ |
| | 31 | INTTM07 | End of timer channel 7 count or capture | | 0048H | | √ | √ | √ | √ |
| | 32 | INTP6 | Pin input edge detection | External | 004AH | (B) | √ | √ | – | – |
| | 33 | INTP7 | | | 004CH | | √ | – | – | – |
| | 34 | INTP8 | | | 004EH | | √ | √ | – | – |
| | 35 | INTP9 | | | 0050H | | √ | √ | – | – |
| | 36 | INTP10 | | | 0052H | | √ | – | – | – |
| | 37 | INTP11 | | | 0054H | | √ | – | – | – |
| <R> | 38 | INTMD | End of division operation/ Overflow of multiply-accumulation result occurs | Internal | 005EH | (A) | √ | √ | √ | √ |
| <R> | 39 | INTFL | Reserved[Note 3] | | 0062H | | √ | √ | √ | √ |
| Software | – | BRK | Execution of BRK instruction | – | 007EH | (D) | √ | √ | √ | √ |
| Reset | – | RESET | $\overline{\text{RESET}}$ pin input | – | 0000H | – | √ | √ | √ | √ |
| | | POR | Power-on-reset | | | | √ | √ | √ | √ |
| | | LVD | Voltage detection[Note 3] | | | | √ | √ | √ | √ |
| | | WDT | Overflow of watchdog timer | | | | √ | √ | √ | √ |
| | | TRAP | Execution of illegal instruction[Note 4] | | | | √ | √ | √ | √ |
| | | IAW | Illegal-memory access | | | | √ | √ | √ | √ |
| | | RPE | RAM parity error | | | | √ | √ | √ | √ |

**Notes 1.** The default priority determines the sequence of interrupts if two or more maskable interrupts occur simultaneously. Zero indicates the highest priority and 39 indicates the lowest priority.

**2.** Basic configuration types (A) to (D) correspond to (A) to (D) in Figure 16-1.

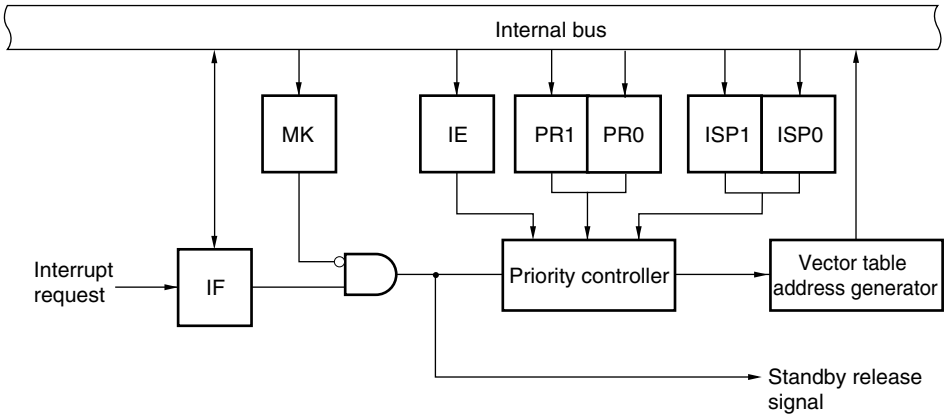
<R> **3.** Be used at the flash self programming library or the data flash library.

**4.** When bit 7 (LVIMD) of the voltage detection level register (LVIS) is set to 1.

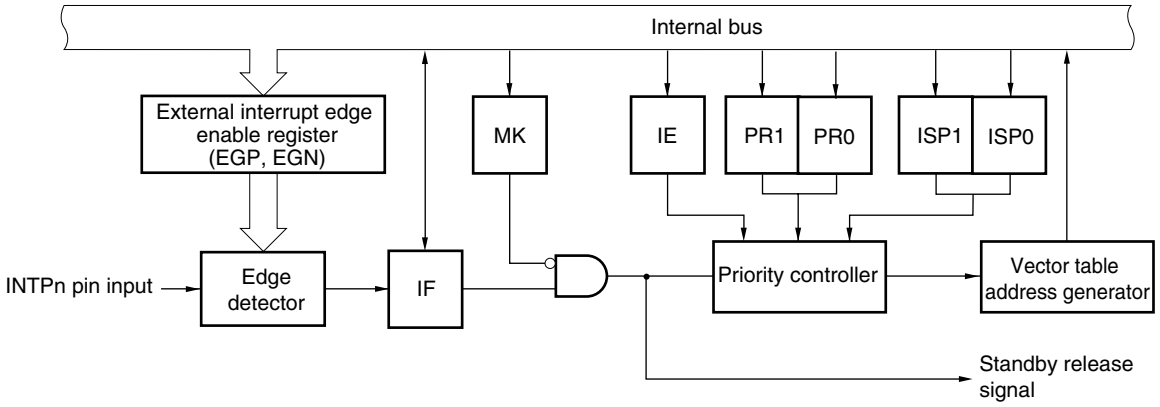
**5.** When the instruction code in FFH is executed.
Reset by the illegal instruction execution not issued by emulation with the in-circuit emulator or on-chip debug emulator.

**Figure 16-1. Basic Configuration of Interrupt Function (1/2)**

**(A) Internal maskable interrupt**




**(B) External maskable interrupt (INTPn)**

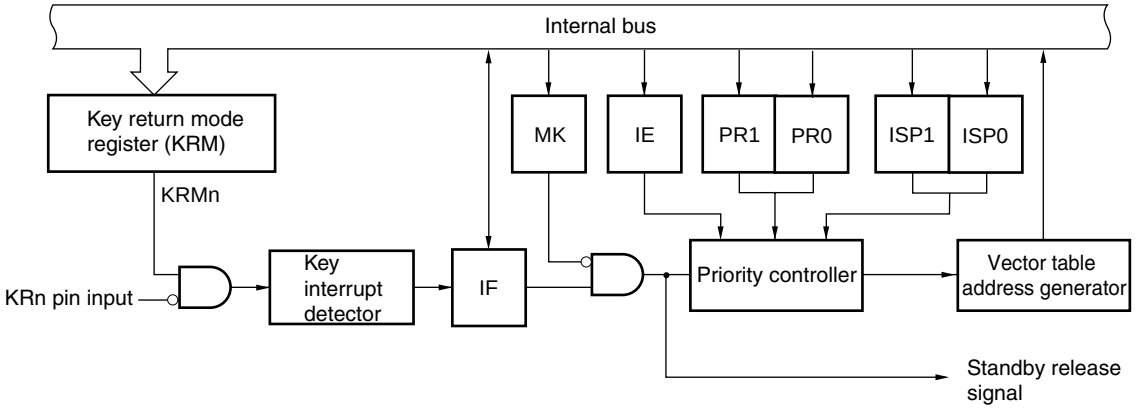



IF: Interrupt request flag
IE: Interrupt enable flag
ISP0: In-service priority flag 0
ISP1: In-service priority flag 1
MK: Interrupt mask flag
PR0: Priority specification flag 0
PR1: Priority specification flag 1

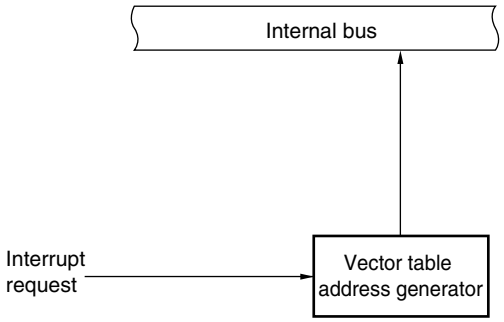
**Remark** 25, 32-pin: n = 0 to 4
48-pin: n = 0 to 6, 8, 9
64-pin: n = 0 to 11

**Figure 16-1. Basic Configuration of Interrupt Function (2/2)**

**(C) External maskable interrupt (INTKR)**




**(D) Software interrupt**




IF: Interrupt request flag
IE: Interrupt enable flag
ISP0: In-service priority flag 0
ISP1: In-service priority flag 1
MK: Interrupt mask flag
PR0: Priority specification flag 0
PR1: Priority specification flag 1

**Remark** 32-pin: n = 0
48-pin: n = 0 to 5
64-pin: n = 0 to 9

## 16.3 Registers Controlling Interrupt Functions

The following 6 types of registers are used to control the interrupt functions.

- Interrupt request flag registers (IF0L, IF0H, IF1L, IF1H, IF2L, IF2H)
- Interrupt mask flag registers (MK0L, MK0H, MK1L, MK1H, MK2L, MK2H)
- Priority specification flag registers (PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR10L, PR10H, PR11L, PR11H, PR12L, PR12H)
- External interrupt rising edge enable registers (EGP0, EGP1)
- External interrupt falling edge enable registers (EGN0, EGN1)
- Program status word (PSW)

Table 16-2 shows a list of interrupt request flags, interrupt mask flags, and priority specification flags corresponding to interrupt request sources.

**Table 16-2. Flags Corresponding to Interrupt Request Sources (1/4)**

| Interrupt Source | Interrupt Request Flag | | Interrupt Mask Flag | | Priority Specification Flag | | 64-pin | 48-pin | 32-pin | 25-pin |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Register | | Register | | Register | | | | |
| INTWDTI | WDTIIF | IF0L | WDTIMK | MK0L | WDTIPR0, WDTIPR1 | PR00L, PR10L | √ | √ | √ | √ |
| INTLVI | LVIIF | | LVIMK | | LVIPR0, LVIPR1 | | √ | √ | √ | √ |
| INTP0 | PIF0 | | PMK0 | | PPR00, PPR10 | | √ | √ | √ | √ |
| INTP1 | PIF1 | | PMK1 | | PPR01, PPR11 | | √ | √ | √ | √ |
| INTP2 | PIF2 | | PMK2 | | PPR02, PPR12 | | √ | √ | √ | √ |
| INTP3 | PIF3 | | PMK3 | | PPR03, PPR13 | | √ | √ | √ | √ |
| INTP4 | PIF4 | | PMK4 | | PPR04, PPR14 | | √ | √ | √ | √ |
| INTP5 | PIF5 | | PMK5 | | PPR05, PPR15 | | √ | √ | – | – |

**Table 16-2. Flags Corresponding to Interrupt Request Sources (2/4)**

| Interrupt Source | Interrupt Request Flag | Register | Interrupt Mask Flag | Register | Priority Specification Flag | Register | 64-pin | 48-pin | 32-pin | 25-pin |
|---|---|---|---|---|---|---|---|---|---|---|
| INTST2[Note 1] | STIF2[Note 1] | IF0H | STMK2[Note 1] | MK0H | STPR02, STPR12[Note 1] | PR00H, PR10H | √ | √ | √ | – |
| INTCSI20[Note 1] | CSIIF20[Note 1] | | CSIMK20[Note 1] | | CSIPR020, CSIPR120[Note 1] | | √ | √ | √ | – |
| INTIIC20[Note 1] | IICIF20[Note 1] | | IICMK20[Note 1] | | IICPR020, IICPR120[Note 1] | | √ | √ | √ | – |
| INTSR2[Note 2] | SRIF2[Note 2] | | SRMK2[Note 2] | | SRPR02, SRPR12[Note 2] | | √ | √ | √ | – |
| INTCSI21[Note 2] | CSIIF21[Note 2] | | CSIMK21[Note 2] | | CSIPR021, CSIPR121[Note 2] | | √ | √ | – | – |
| INTIIC21[Note 2] | IICIF21[Note 2] | | IICMK21[Note 2] | | IICPR021, IICPR121[Note 2] | | √ | √ | – | – |
| INTSRE2 | SREIF2 | | SREMK2 | | SREPR02, SREPR12 | | √ | √ | √ | – |
| INTDMA0 | DMAIF0 | | DMAMK0 | | DMAPR00, DMAPR10 | | √ | √ | √ | √ |
| INTDMA1 | DMAIF1 | | DMAMK1 | | DMAPR01, DMAPR11 | | √ | √ | √ | √ |
| INTST0[Note 3] | STIF0[Note 3] | | STMK0[Note 3] | | STPR00, STPR10[Note 3] | | √ | √ | √ | √ |
| INTCSI00[Note 3] | CSIIF00[Note 3] | | CSIMK00[Note 3] | | CSIPR000, CSIPR100[Note 3] | | √ | √ | √ | √ |
| INTIIC00[Note 3] | IICIF00[Note 3] | | IICMK00[Note 3] | | IICPR000, IICPR100[Note 3] | | √ | √ | √ | √ |
| INTSR0[Note 4] | SRIF0[Note 4] | | SRMK0[Note 4] | | SRPR00, SRPR10[Note 4] | | √ | √ | √ | √ |
| INTCSI01[Note 4] | CSIIF01[Note 4] | | CSIMK01[Note 4] | | CSIPR001, CSIPR101[Note 4] | | √ | √ | – | – |
| INTIIC01[Note 4] | IICIF01[Note 4] | | IICMK01[Note 4] | | IICPR001, IICPR101[Note 4] | | √ | √ | – | – |
| INTSRE0[Note 5] | SREIF0[Note 5] | | SREMK0[Note 5] | | SREPR00, SREPR10[Note 5] | | √ | √ | √ | √ |
| INTTM01H[Note 5] | TMIF01H[Note 5] | | TMMK01H[Note 5] | | TMPR001H, TMPR101H[Note 5] | | √ | √ | √ | √ |

**Notes 1.** If one of the interrupt sources INTST2, INTCSI20, and INTIIC20 is generated, bit 0 of the IF0H register is set to 1. Bit 0 of the MK0H, PR00H, and PR10H registers can be used for all three of these interrupt sources.

**2.** If one of the interrupt sources INTSR2, INTCSI21, and INTIIC21 is generated, bit 1 of the IF0H register is set to 1. Bit 1 of the MK0H, PR00H, and PR10H registers can be used for all three of these interrupt sources.

**3.** If one of the interrupt sources INTST0, INTCSI00, and INTIIC00 is generated, bit 5 of the IF0H register is set to 1. Bit 5 of the MK0H, PR00H, and PR10H registers can be used for all three of these interrupt sources.

**4.** If one of the interrupt sources INTSR0, INTCSI01, and INTIIC01 is generated, bit 6 of the IF0H register is set to 1. Bit 6 of the MK0H, PR00H, and PR10H registers can be used for all three of these interrupt sources.

**5.** Do not use the error interrupt of UART0 reception and the interrupt of channel 1 of TAU0 (while the higher 8 bits are operating at a timer) at the same time because they share flags for the interrupt request sources. If the error interrupt of UART0 reception is not used (EOC01 = 0), UART0 and channel 1 of TAU0 (while the higher 8 bits are operating at a timer) can be used at the same time. If the interrupt source INTSRE0 or INTTM01H is generated, bit 7 of the IF0H register is set to 1. Bit 7 of the MK0H, PR00H, and PR10H registers can be used for both these interrupt sources.

**Table 16-2. Flags Corresponding to Interrupt Request Sources (3/4)**

| Interrupt Source | Interrupt Request Flag | Register | Interrupt Mask Flag | Register | Priority Specification Flag | Register | 64-pin | 48-pin | 32-pin | 25-pin |
|---|---|---|---|---|---|---|---|---|---|---|
| INTST1 Note 1 | STIF1 Note 1 | IF1L | STMK1 Note 1 | MK1L | STPR01, STPR11 Note 1 | PR01L, PR11L | √ | √ | √ | √ |
| INTCSI10 Note 1 | CSIIF10 Note 1 | | CSIMK10 Note 1 | | CSIPR010, CSIPR110 Note 1 | | √ | – | – | – |
| INTIIC10 Note 1 | IICIF10 Note 1 | | IICMK10 Note 1 | | IICPR010, IICPR110 Note 1 | | √ | – | – | – |
| INTSR1 Note 2 | SRIF1 Note 2 | | SRMK1 Note 2 | | SRPR01, SRPR11 Note 2 | | √ | √ | √ | √ |
| INTCSI11 Note 2 | CSIIF11 Note 2 | | CSIMK11 Note 2 | | CSIPR011, CSIPR111 Note 2 | | √ | √ | √ | √ |
| INTIIC11 Note 2 | IICIF11 Note 2 | | IICMK11 Note 2 | | IICPR011, IICPR111 Note 2 | | √ | √ | √ | √ |
| INTSRE1 Note 3 | SREIF1 Note 3 | | SREMK1 Note 3 | | SREPR01, SREPR11 Note 3 | | √ | √ | √ | √ |
| INTTM03H Note 3 | TMIF03H Note 3 | | TMMK03H Note 3 | | TMPR003H, TMPR103H Note 3 | | √ | √ | √ | √ |
| INTIICA0 | IICAIF0 | | IICAMK0 | | IICAPR00, IICAPR10 | | √ | √ | √ | √ |
| INTTM00 | TMIF00 | | TMMK00 | | TMPR000, TMPR100 | | √ | √ | √ | √ |
| INTTM01 | TMIF01 | | TMMK01 | | TMPR001, TMPR101 | | √ | √ | √ | √ |
| INTTM02 | TMIF02 | | TMMK02 | | TMPR002, TMPR102 | | √ | √ | √ | √ |
| INTTM03 | TMIF03 | | TMMK03 | | TMPR003, TMPR103 | | √ | √ | √ | √ |
| INTAD | ADIF | IF1H | ADMK | MK1H | ADPR0, ADPR1 | PR01H, PR11H | √ | √ | √ | √ |
| INTRTC | RTCIF | | RTCMK | | RTCPR0, RTCPR1 | | √ | √ | √ | √ |
| INTIT | ITIF | | ITMK | | ITPR0, ITPR1 | | √ | √ | √ | √ |
| INTKR | KRIF | | KRMK | | KRPR0, KRPR1 | | √ | √ | √ | (√) |
| INTTM04 | TMIF04 | | TMMK04 | | TMPR004, TMPR104 | | √ | √ | √ | √ |

**Notes 1.** If one of the interrupt sources INTST1, INTCSI10, and INTIIC10 is generated, bit 0 of the IF1L register is set to 1. Bit 0 of the MK1L, PR01L, and PR11L registers can be used for all three of these interrupt sources.

**2.** If one of the interrupt sources INTSR1, INTCSI11, and INTIIC11 is generated, bit 1 of the IF1L register is set to 1. Bit 1 of the MK1L, PR01L, and PR11L registers can be used for all three of these interrupt sources.

**3.** Do not use the error interrupt of UART1 reception and the interrupt of channel 3 of TAU0 (while the higher 8 bits are operating at a timer) at the same time because they share flags for the interrupt request sources. If <R> the error interrupt of UART1 reception is not used (EOC03 = 0), UART1 and channel 3 of TAU0 (while the higher 8 bits are operating at a timer) can be used at the same time. If the interrupt source INTSRE1 or INTTM03H is generated, bit 2 of the IF1L register is set to 1. Bit 2 of the MK1L, PR01L, and PR11L registers can be used for both these interrupt sources.

**Table 16-2. Flags Corresponding to Interrupt Request Sources (4/4)**

| Interrupt Source | Interrupt Request Flag | | Interrupt Mask Flag | | Priority Specification Flag | | 64-pin | 48-pin | 32-pin | 25-pin |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Register | | Register | | Register | | | | |
| INTTM05 | TMIF05 | IF2L | TMMK05 | MK2L | TMPR005, TMPR105 | PR02L, PR12L | √ | √ | √ | √ |
| INTTM06 | TMIF06 | | TMMK06 | | TMPR006, TMPR106 | | √ | √ | √ | √ |
| INTTM07 | TMIF07 | | TMMK07 | | TMPR007, TMPR107 | | √ | √ | √ | √ |
| INTP6 | PIF6 | | PMK6 | | PPR06, PPR16 | | √ | √ | – | – |
| INTP7 | PIF7 | | PMK7 | | PPR07, PPR17 | | √ | – | – | – |
| INTP8 | PIF8 | | PMK8 | | PPR08, PPR18 | | √ | √ | – | – |
| INTP9 | PIF9 | | PMK9 | | PPR09, PPR19 | | √ | √ | – | – |
| INTP10 | PIF10 | | PMK10 | | PPR010, PPR110 | | √ | – | – | – |
| INTP11 | PIF11 | IF2H | PMK11 | MK2H | PPR011, PPR111 | PR02H, PR12H | √ | – | – | – |
| INTMD | MDIF | | MDMK | | MDPR0, MDPR1 | | √ | √ | √ | √ |
| INTFL | FLIF | | FLMK | | FLPR0, FLPR1 | | √ | √ | √ | √ |

### 16.3.1 Interrupt request flag registers (IF0L, IF0H, IF1L, IF1H, IF2L, IF2H)

The interrupt request flags are set to 1 when the corresponding interrupt request is generated or an instruction is executed. They are cleared to 0 when an instruction is executed upon acknowledgment of an interrupt request or upon reset signal generation.

When an interrupt is acknowledged, the interrupt request flag is automatically cleared and then the interrupt routine is entered.

The IF0L, IF0H, IF1L, IF1H, IF2L, and IF2H registers can be set by a 1-bit or 8-bit memory manipulation instruction. When the IF0L and IF0H registers, the IF1L and IF1H registers, and the IF2L and IF2H registers are combined to form 16-bit registers IF0, IF1, and IF2, they can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H.

**Remark** If an instruction that writes data to this register is executed, the number of instruction execution clocks increases by 2 clocks.

<R> **Figure 16-2. Format of Interrupt Request Flag Registers (IF0L, IF0H, IF1L, IF1H, IF2L, IF2H) (1/2)**

Address: FFFE0H After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| IF0L | PIF5 | PIF4 | PIF3 | PIF2 | PIF1 | PIF0 | LVIIF | WDTIIF |

Address: FFFE1H After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| IF0H | SREIF0<br>TMIF01H | SRIF0<br>CSIIF01<br>IICIF01 | STIF0<br>CSIIF00<br>IICIF00 | DMAIF1 | DMAIF0 | SREIF2<br>TMIF11H | SRIF2<br>CSIIF21<br>IICIF21 | STIF2<br>CSIIF20<br>IICIF20 |

Address: FFFE2H After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| IF1L | TMIF03 | TMIF02 | TMIF01 | TMIF00 | IICAIF0 | SREIF1<br>TMIF03H | SRIF1<br>CSIIF11<br>IICIF11 | STIF1<br>CSIIF10<br>IICIF10 |

Address: FFFE3H After reset: 00H R/W

| Symbol | <7> | 6 | 5 | 4 | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| IF1H | TMIF04 | 0 | 0 | 0 | KRIF | ITIF | RTCIF | ADIF |

Address: FFFD0H After reset: 00H R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| IF2L | PIF10 | PIF9 | PIF8 | PIF7 | PIF6 | TMIF07 | TMIF06 | TMIF05 |

<R> **Figure 16-2. Format of Interrupt Request Flag Registers (IF0L, IF0H, IF1L, IF1H, IF2L, IF2H) (2/2)**

Address: FFFD1H After reset: 00H R/W

| Symbol | <7> | 6 | <5> | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| IF2H | FLIF | 0 | MDIF | 0 | 0 | 0 | 0 | PIF11 |

| XXIFX | Interrupt request flag |
|---|---|
| 0 | No interrupt request signal is generated |
| 1 | Interrupt request is generated, interrupt request status |

<R> **Cautions 1. The available registers and bits differ depending on the product. For details about the registers and bits available for each product, see Table 16-2. Be sure to clear bits that are not available to the initial value.**

**2. When manipulating a flag of the interrupt request flag register, use a 1-bit memory manipulation instruction (CLR1). When describing in C language, use a bit manipulation instruction such as "IF0L.0 = 0;" or "_asm("clr1 IF0L, 0");" because the compiled assembler must be a 1-bit memory manipulation instruction (CLR1).**

**If a program is described in C language using an 8-bit memory manipulation instruction such as "IF0L &= 0xfe;" and compiled, it becomes the assembler of three instructions.**

```
mov a, IF0L
and a, #0FEH
mov IF0L, a
```

**In this case, even if the request flag of the another bit of the same interrupt request flag register (IF0L) is set to 1 at the timing between "mov a, IF0L" and "mov IF0L, a", the flag is cleared to 0 at "mov IF0L, a". Therefore, care must be exercised when using an 8-bit memory manipulation instruction in C language.**

### 16.3.2 Interrupt mask flag registers (MK0L, MK0H, MK1L, MK1H, MK2L, MK2H)

<R> The interrupt mask flags are used to enable/disable the corresponding maskable interrupt.

The MK0L, MK0H, MK1L, MK1H, MK2L, and MK2H registers can be set by a 1-bit or 8-bit memory manipulation instruction. When the MK0L and MK0H registers, the MK1L and MK1H registers, and the MK2L and MK2H registers are combined to form 16-bit registers MK0, MK1, and MK2, they can be set by a 16-bit memory manipulation instruction.

Reset signal generation sets these registers to FFH.

**Remark** If an instruction that writes data to this register is executed, the number of instruction execution clocks increases by 2 clocks.

<R> **Figure 16-3. Format of Interrupt Mask Flag Registers (MK0L, MK0H, MK1L, MK1H, MK2L, MK2H)**

Address: FFFE4H After reset: FFH R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| MK0L | PMK5 | PMK4 | PMK3 | PMK2 | PMK1 | PMK0 | LVIMK | WDTIMK |

Address: FFFE5H After reset: FFH R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| MK0H | SREMK0<br>TMMK01H | SRMK0<br>CSIMK01<br>IICMK01 | STMK0<br>CSIMK00<br>IICMK00 | DMAMK1 | DMAMK0 | SREMK2<br>TMMK11H | SRMK2<br>CSIMK21<br>IICMK21 | STMK2<br>CSIMK20<br>IICMK20 |

Address: FFFE6H After reset: FFH R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| MK1L | TMMK03 | TMMK02 | TMMK01 | TMMK00 | IICAMK0 | SREMK1<br>TMMK03H | SRMK1<br>CSIMK11<br>IICMK11 | STMK1<br>CSIMK10<br>IICMK10 |

Address: FFFE7H After reset: FFH R/W

| Symbol | <7> | 6 | 5 | 4 | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| MK1H | TMMK04 | 1 | 1 | 1 | KRMK | ITMK | RTCMK | ADMK |

Address: FFFD4H After reset: FFH R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| MK2L | PMK10 | PMK9 | PMK8 | PMK7 | PMK6 | TMMK07 | TMMK06 | TMMK05 |

Address: FFFD5H After reset: FFH R/W

| Symbol | <7> | 6 | <5> | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| MK2H | FLMK | 1 | MDMK | 1 | 1 | 1 | 1 | PMK11 |

| XXMKX | Interrupt servicing control |
|---|---|
| 0 | Interrupt servicing enabled |
| 1 | Interrupt servicing disabled |

<R> **Caution The available registers and bits differ depending on the product. For details about the registers and bits available for each product, see Table 16-2 Be sure to set bits that are not available to the initial value.**

### 16.3.3 Priority specification flag registers (PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR10L, PR10H, PR11L, PR11H, PR12L, PR12H)

The priority specification flag registers are used to set the corresponding maskable interrupt priority level.

A priority level is set by using the PR0xy and PR1xy registers in combination (xy = 0L, 0H, 1L, 1H, 2L, or 2H).

The PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR10L, PR10H, PR11L, PR11H, PR12L, and the PR12H registers can be set by a 1-bit or 8-bit memory manipulation instruction. If the PR00L and PR00H registers, the PR01L and PR01H registers, the PR02L and PR02H registers, the PR10L and PR10H registers, the PR11L and PR11H registers, and the PR12L and PR12H registers are combined to form 16-bit registers PR00, PR01, PR02, PR10, PR11, and PR12, they can be set by a 16-bit memory manipulation instruction.

Reset signal generation sets these registers to FFH.

**Remark** If an instruction that writes data to this register is executed, the number of instruction execution clocks increases by 2 clocks.

<R> **Figure 16-4. Format of Priority Specification Flag Registers (PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR10L, PR10H, PR11L, PR11H, PR12L, PR12H) (1/2)**

Address: FFFE8H After reset: FFH R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PR00L | PPR05 | PPR04 | PPR03 | PPR02 | PPR01 | PPR00 | LVIPR0 | WDTIPR0 |

Address: FFFECH After reset: FFH R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PR10L | PPR15 | PPR14 | PPR13 | PPR12 | PPR11 | PPR10 | LVIPR1 | WDTIPR1 |

Address: FFFE9H After reset: FFH R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PR00H | SREPR00<br>TMPR001H | SRPR00<br>CSIPR001<br>IICPR001 | STPR00<br>CSIPR000<br>IICPR000 | DMAPR01 | DMAPR00 | SREPR02<br>TMPR011H | SRPR02<br>CSIPR021<br>IICPR021 | STPR02<br>CSIPR020<br>IICPR020 |

Address: FFFEDH After reset: FFH R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PR10H | SREPR10<br>TMPR101H | SRPR10<br>CSIPR101<br>IICPR101 | STPR10<br>CSIPR100<br>IICPR100 | DMAPR11 | DMAPR10 | SREPR12<br>TMPR111H | SRPR12<br>CSIPR121<br>IICPR121 | STPR12<br>CSIPR120<br>IICPR120 |

Address: FFFEAH After reset: FFH R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PR01L | TMPR003 | TMPR002 | TMPR001 | TMPR000 | IICAPR00 | SREPR01<br>TMPR003H | SRPR01<br>CSIPR011<br>IICPR011 | STPR01<br>CSIPR010<br>IICPR010 |

Address: FFFEEH After reset: FFH R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PR11L | TMPR103 | TMPR102 | TMPR101 | TMPR100 | IICAPR10 | SREPR11<br>TMPR103H | SRPR11<br>CSIPR111<br>IICPR111 | STPR11<br>CSIPR110<br>IICPR110 |

<R> **Figure 16-4. Format of Priority Specification Flag Registers (PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR10L, PR10H, PR11L, PR11H, PR12L, PR12H) (2/2)**

Address: FFFEBH After reset: FFH R/W

| Symbol | <7> | 6 | 5 | 4 | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PR01H | TMPR004 | 1 | 1 | 1 | KRPR0 | ITPR0 | RTCPR0 | ADPR0 |

Address: FFFEFH After reset: FFH R/W

| Symbol | <7> | 6 | 5 | 4 | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PR11H | TMPR104 | 1 | 1 | 1 | KRPR1 | ITPR1 | RTCPR1 | ADPR1 |

Address: FFFD8H After reset: FFH R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PR02L | PPR010 | PPR09 | PPR08 | PPR07 | PPR06 | TMPR007 | TMPR006 | TMPR005 |

Address: FFFDCH After reset: FFH R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| PR12L | PPR110 | PPR19 | PPR18 | PPR17 | PPR16 | TMPR107 | TMPR106 | TMPR105 |

Address: FFFD9H After reset: FFH R/W

| Symbol | <7> | 6 | <5> | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PR02H | FLPR0 | 1 | MDPR0 | 1 | 1 | 1 | 1 | PPR011 |

Address: FFFDDH After reset: FFH R/W

| Symbol | <7> | 6 | <5> | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| PR12H | FLPR1 | 1 | MDPR1 | 1 | 1 | 1 | 1 | PPR111 |

| XXPR1X | XXPR0X | Priority level selection |
|---|---|---|
| 0 | 0 | Specify level 0 (high priority level) |
| 0 | 1 | Specify level 1 |
| 1 | 0 | Specify level 2 |
| 1 | 1 | Specify level 3 (low priority level) |

<R> **Caution The available registers and bits differ depending on the product. For details about the registers and bits available for each product, see Table 16-2 Be sure to set bits that are not available to the initial value.**

### 16.3.4 External interrupt rising edge enable registers (EGP0, EGP1), external interrupt falling edge enable registers (EGN0, EGN1)

These registers specify the valid edge for INTP0 to INTP11.

The EGP0, EGP1, EGN0, and EGN1 registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H.

<R> **Figure 16-5. Format of External Interrupt Rising Edge Enable Registers (EGP0, EGP1) and External Interrupt Falling Edge Enable Registers (EGN0, EGN1)**

Address: FFF38H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| EGP0 | EGP7 | EGP6 | EGP5 | EGP4 | EGP3 | EGP2 | EGP1 | EGP0 |

Address: FFF39H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| EGN0 | EGN7 | EGN6 | EGN5 | EGN4 | EGN3 | EGN2 | EGN1 | EGN0 |

Address: FFF3AH After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| EGP1 | 0 | 0 | 0 | 0 | EGP11 | EGP10 | EGP9 | EGP8 |

Address: FFF3BH After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| EGN1 | 0 | 0 | 0 | 0 | EGN11 | EGN10 | EGN9 | EGN8 |

| EGPn | EGNn | INTPn pin valid edge selection (n = 0 to 11) |
|---|---|---|
| 0 | 0 | Edge detection disabled |
| 0 | 1 | Falling edge |
| 1 | 0 | Rising edge |
| 1 | 1 | Both rising and falling edges |

Table 16-3 shows the ports corresponding to the EGPn and EGNn bits.

**Table 16-3. Interrupt Request Signal Corresponding to EGPn and EGNn bits**

| Detection Enable Bit | | Edge Detection Port | Interrupt Request Signal | 64-pin | 48-pin | 32-pin | 25-pin |
|---|---|---|---|---|---|---|---|
| EGP0 | EGN0 | P137 | INTP0 | √ | √ | √ | √ |
| EGP1 | EGN1 | P50 | INTP1 | √ | √ | √ | √ |
| EGP2 | EGN2 | P51 | INTP2 | √ | √ | √ | √ |
| EGP3 | EGN3 | P30 | INTP3 | √ | √ | √ | √ |
| EGP4 | EGN4 | P31 | INTP4 | √ | √ | √ | √ |
| EGP5 | EGN5 | P16 | INTP5 | √ | √ | – | – |
| EGP6 | EGN6 | P140 | INTP6 | √ | √ | – | – |
| EGP7 | EGN7 | P141 | INTP7 | √ | – | – | – |
| EGP8 | EGN8 | P74 | INTP8 | √ | √ | – | – |
| EGP9 | EGN9 | P75 | INTP9 | √ | √ | – | – |
| EGP10 | EGN10 | P76 | INTP10 | √ | – | – | – |
| EGP11 | EGN11 | P77 | INTP11 | √ | – | – | – |

**Caution When the input port pins used for the external interrupt functions are switched to the output mode, the INTPn interrupt might be generated upon detection of a valid edge.**
**When switching the input port pins to the output mode, set the port mode register (PMxx) to 0 after disabling the edge detection (by setting EGPn and EGNn to 0).**

**Remarks 1.** For edge detection port, see **2.1 Port Function**.
**2.** n = 0 to 11

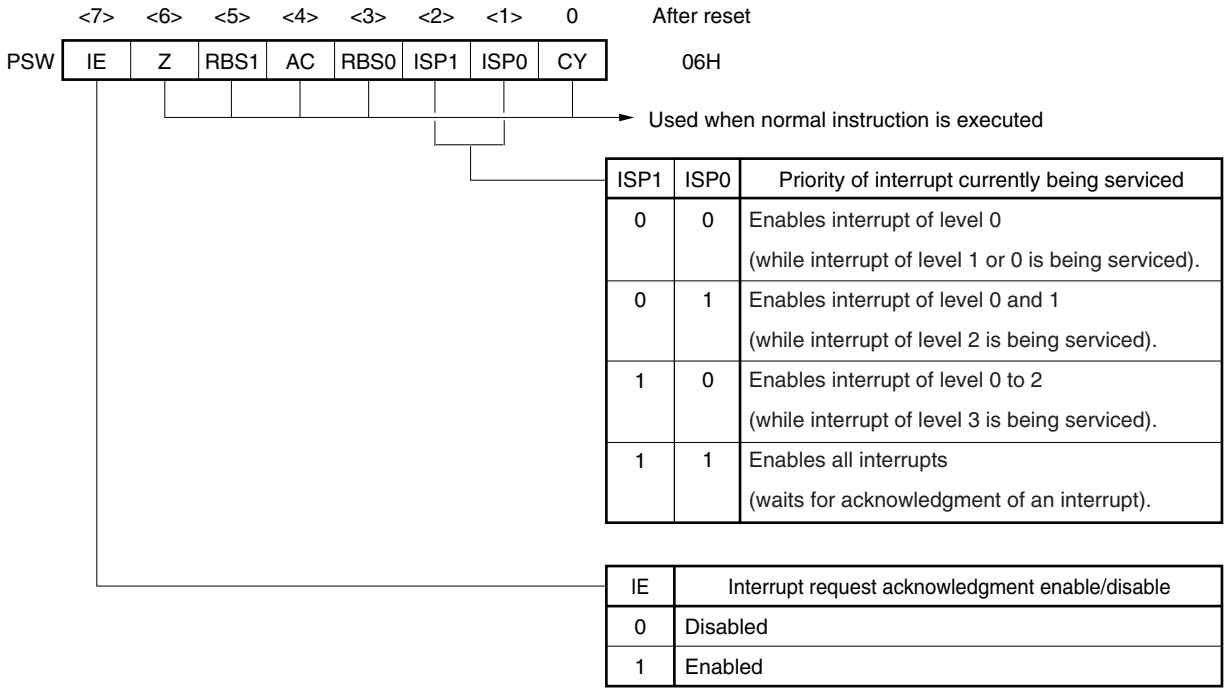
### <R> 16.3.5 Program status word (PSW)

The program status word is a register used to hold the instruction execution result and the current status for an interrupt request. The IE flag that sets maskable interrupt enable/disable and the ISP0 and ISP1 flags that controls multiple interrupt servicing are mapped to the PSW.

Besides 8-bit read/write, this register can carry out operations using bit manipulation instructions and dedicated instructions (EI and DI). When a vectored interrupt request is acknowledged, if the BRK instruction is executed, the contents of the PSW are automatically saved into a stack and the IE flag is reset to 0. If a maskable interrupt request is acknowledged, the contents of the priority specification flag register of the acknowledged interrupt are not 00, its value minus 1 is transferred to the ISP0 and ISP1 flags. The PSW contents are also saved into the stack with the PUSH PSW instruction. They are restored from the stack with the RETI, RETB, and POP PSW instructions.

Reset signal generation sets PSW to 06H.

**Figure 16-6. Configuration of Program Status Word**




| ISP1 | ISP0 | Priority of interrupt currently being serviced |
|---|---|---|
| 0 | 0 | Enables interrupt of level 0<br>(while interrupt of level 1 or 0 is being serviced). |
| 0 | 1 | Enables interrupt of level 0 and 1<br>(while interrupt of level 2 is being serviced). |
| 1 | 0 | Enables interrupt of level 0 to 2<br>(while interrupt of level 3 is being serviced). |
| 1 | 1 | Enables all interrupts<br>(waits for acknowledgment of an interrupt). |

| IE | Interrupt request acknowledgment enable/disable |
|---|---|
| 0 | Disabled |
| 1 | Enabled |

## 16.4 Interrupt Servicing Operations

### 16.4.1 Maskable interrupt request acknowledgment

A maskable interrupt request becomes acknowledgeable when the interrupt request flag is set to 1 and the mask (MK) flag corresponding to that interrupt request is cleared to 0. A vectored interrupt request is acknowledged if interrupts are in the interrupt enabled state (when the IE flag is set to 1). However, a low-priority interrupt request is not acknowledged during servicing of a higher priority interrupt request.

The times from generation of a maskable interrupt request until vectored interrupt servicing is performed are listed in Table 16-4 below.

For the interrupt request acknowledgment timing, see **Figures 16-8** and **16-9**.

**Table 16-4. Time from Generation of Maskable Interrupt Until Servicing**

| | Minimum Time | Maximum Time[Note] |
|---|---|---|
| Servicing time | 9 clocks | 16 clocks |

**Note** Maximum time does not apply when an instruction from the internal RAM area is executed.

**Remark** 1 clock: $1/f_{CLK}$ ($f_{CLK}$: CPU clock)

If two or more maskable interrupt requests are generated simultaneously, the request with a higher priority level specified in the priority specification flag is acknowledged first. If two or more interrupts requests have the same priority level, the request with the highest default priority is acknowledged first.

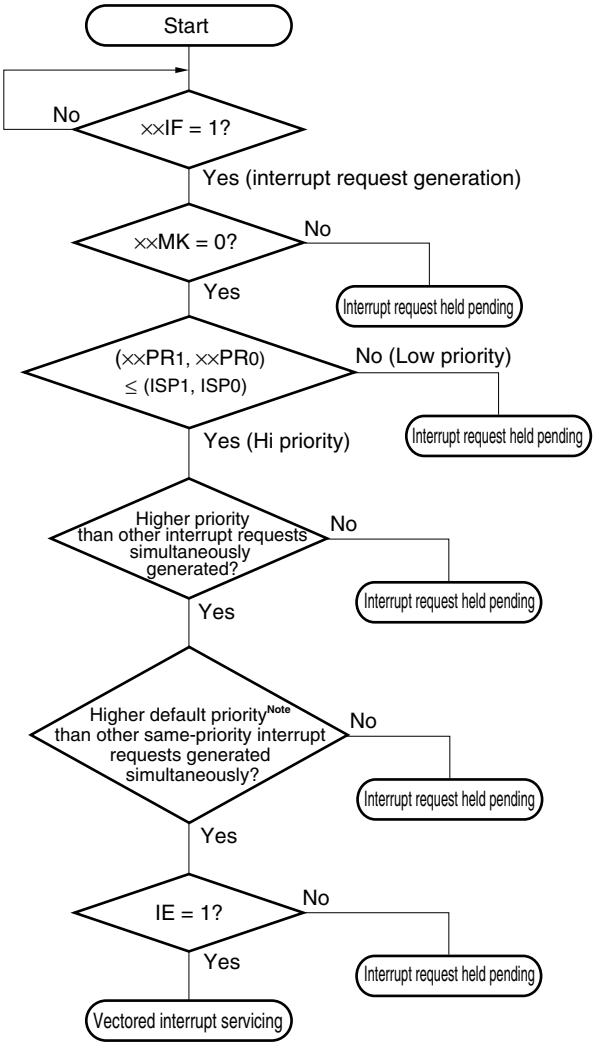
An interrupt request that is held pending is acknowledged when it becomes acknowledgeable.

Figure 16-7 shows the interrupt request acknowledgment algorithm.

If a maskable interrupt request is acknowledged, the contents are saved into the stacks in the order of PSW, then PC, the IE flag is reset (0), and the contents of the priority specification flag corresponding to the acknowledged interrupt are transferred to the ISP1 and ISP0 flags. The vector table data determined for each interrupt request is the loaded into the PC and branched.

Restoring from an interrupt is possible by using the RETI instruction.

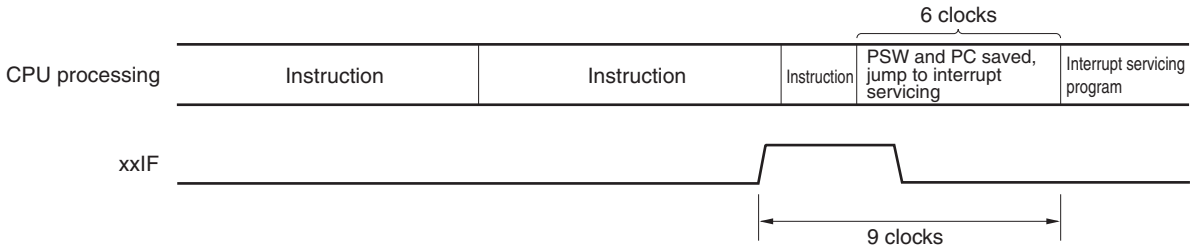
**Figure 16-7. Interrupt Request Acknowledgment Processing Algorithm**




××IF: Interrupt request flag
××MK: Interrupt mask flag
××PR0: Priority specification flag 0
××PR1: Priority specification flag 1
IE: Flag that controls acknowledgment of maskable interrupt request (1 = Enable, 0 = Disable)
ISP0, ISP1: Flag that indicates the priority level of the interrupt currently being serviced (see **Figure 16-6**)

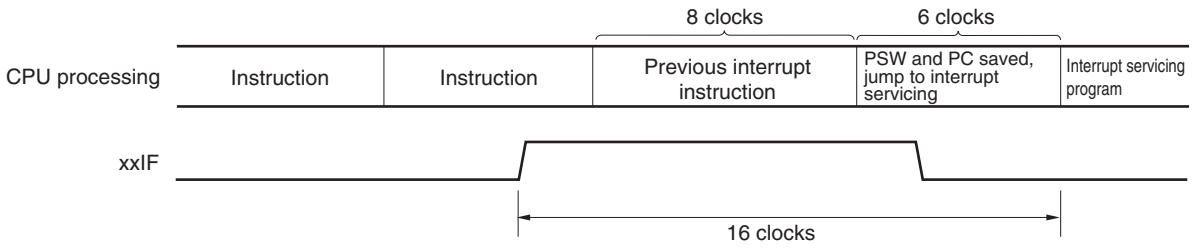
**Note** For the default priority, see **Table 16-1 Interrupt Source List**.

<R> **Figure 16-8. Interrupt Request Acknowledgment Timing (Minimum Time)**




**Remark** 1 clock: 1/$f_{CLK}$ ($f_{CLK}$: CPU clock)

<R> **Figure 16-9. Interrupt Request Acknowledgment Timing (Maximum Time)**




**Remark** 1 clock: 1/$f_{CLK}$ ($f_{CLK}$: CPU clock)

### 16.4.2 Software interrupt request acknowledgment

A software interrupt request is acknowledged by BRK instruction execution. Software interrupts cannot be disabled.

If a software interrupt request is acknowledged, the contents are saved into the stacks in the order of the program status word (PSW), then program counter (PC), the IE flag is reset (0), and the contents of the vector table (0007EH, 0007FH) are loaded into the PC and branched.

Restoring from a software interrupt is possible by using the RETB instruction.

**Caution Can not use the RETI instruction for restoring from the software interrupt.**

### 16.4.3 Multiple interrupt servicing

Multiple interrupt servicing occurs when another interrupt request is acknowledged during execution of an interrupt.

Multiple interrupt servicing does not occur unless the interrupt request acknowledgment enabled state is selected (IE = 1). When an interrupt request is acknowledged, interrupt request acknowledgment becomes disabled (IE = 0). Therefore, to enable multiple interrupt servicing, it is necessary to set (1) the IE flag with the EI instruction during interrupt servicing to enable interrupt acknowledgment.

Moreover, even if interrupts are enabled, multiple interrupt servicing may not be enabled, this being subject to interrupt priority control. Two types of priority control are available: default priority control and programmable priority control. Programmable priority control is used for multiple interrupt servicing.

In the interrupt enabled state, if an interrupt request with a priority higher than that of the interrupt currently being serviced is generated, it is acknowledged for multiple interrupt servicing. If an interrupt with a priority equal to or lower than that of the interrupt currently being serviced is generated during interrupt servicing, it is not acknowledged for multiple interrupt servicing. Interrupt requests that are not enabled because interrupts are in the interrupt disabled state or because they have a lower priority are held pending. When servicing of the current interrupt ends, the pending interrupt request is acknowledged following execution of at least one main processing instruction execution.

Table 16-5 shows relationship between interrupt requests enabled for multiple interrupt servicing and Figure 16-10 shows multiple interrupt servicing examples.

**Table 16-5. Relationship Between Interrupt Requests Enabled for Multiple Interrupt Servicing During Interrupt Servicing**

| Multiple Interrupt Request / Interrupt Being Serviced | | Maskable Interrupt Request | | | | | | | | Software Interrupt Request |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Priority Level 0 (PR = 00) | | Priority Level 1 (PR = 01) | | Priority Level 2 (PR = 10) | | Priority Level 3 (PR = 11) | | |
| | | IE = 1 | IE = 0 | IE = 1 | IE = 0 | IE = 1 | IE = 0 | IE = 1 | IE = 0 | |
| Maskable interrupt | ISP1 = 0, ISP0 = 0 | ○ | × | × | × | × | × | × | × | ○ |
| | ISP1 = 0, ISP0 = 1 | ○ | × | ○ | × | × | × | × | × | ○ |
| | ISP1 = 1, ISP0 = 0 | ○ | × | ○ | × | ○ | × | × | × | ○ |
| <R> | ISP1 = 1, ISP0 = 1 | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| Software interrupt | | ○ | × | ○ | × | ○ | × | ○ | × | ○ |

**Remarks 1.** ○: Multiple interrupt servicing enabled

**2.** ×: Multiple interrupt servicing disabled

**3.** ISP0, ISP1, and IE are flags contained in the PSW.

ISP1 = 0, ISP0 = 0: An interrupt of level 1 or level 0 is being serviced.

ISP1 = 0, ISP0 = 1: An interrupt of level 2 is being serviced.

ISP1 = 1, ISP0 = 0: An interrupt of level 3 is being serviced.

<R> ISP1 = 1, ISP0 = 1: Wait for An interrupt acknowledgment (all interrupts are enabled).

IE = 0: Interrupt request acknowledgment is disabled.

IE = 1: Interrupt request acknowledgment is enabled.

**4.** PR is a flag contained in the PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR10L, PR10H, PR11L, PR11H, PR12L, and PR12H registers.

PR = 00: Specify level 0 with ××PR1× = 0, ××PR0× = 0 (higher priority level)

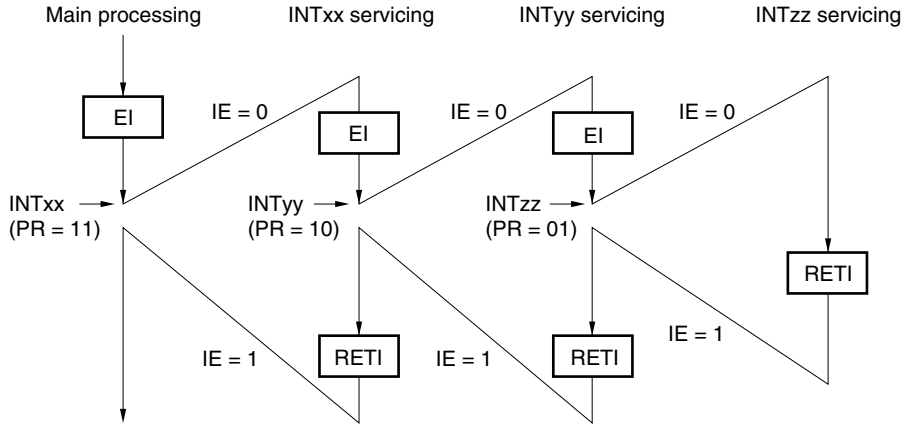
PR = 01: Specify level 1 with ××PR1× = 0, ××PR0× = 1

PR = 10: Specify level 2 with ××PR1× = 1, ××PR0× = 0

PR = 11: Specify level 3 with ××PR1× = 1, ××PR0× = 1 (lower priority level)

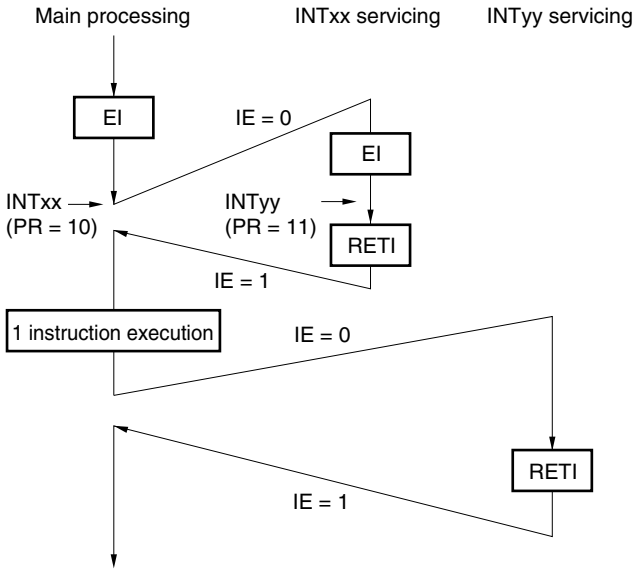
**Figure 16-10. Examples of Multiple Interrupt Servicing (1/2)**

**Example 1. Multiple interrupt servicing occurs twice**




During servicing of interrupt INTxx, two interrupt requests, INTyy and INTzz, are acknowledged, and multiple interrupt servicing takes place. Before each interrupt request is acknowledged, the EI instruction must always be issued to enable interrupt request acknowledgment.

**Example 2. Multiple interrupt servicing does not occur due to priority control**

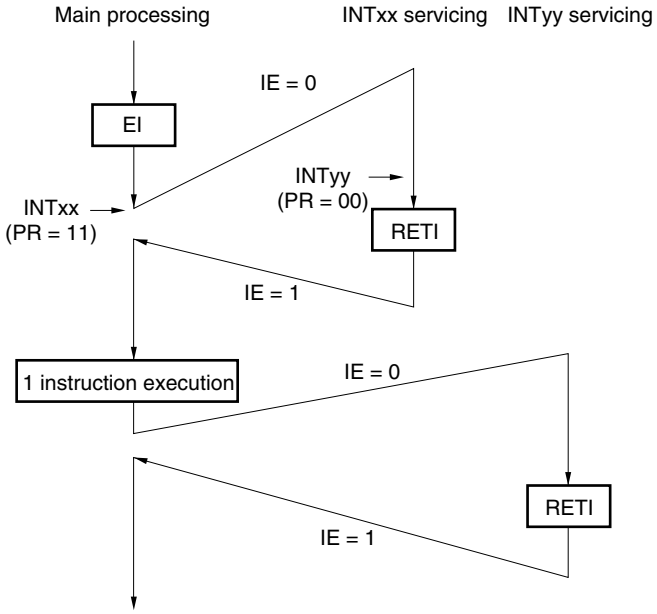



Interrupt request INTyy issued during servicing of interrupt INTxx is not acknowledged because its priority is lower than that of INTxx, and multiple interrupt servicing does not take place. The INTyy interrupt request is held pending, and is acknowledged following execution of one main processing instruction.

PR = 00: Specify level 0 with ××PR1× = 0, ××PR0× = 0 (higher priority level)
PR = 01: Specify level 1 with ××PR1× = 0, ××PR0× = 1
PR = 10: Specify level 2 with ××PR1× = 1, ××PR0× = 0
PR = 11: Specify level 3 with ××PR1× = 1, ××PR0× = 1 (lower priority level)
IE = 0: Interrupt request acknowledgment is disabled
IE = 1: Interrupt request acknowledgment is enabled.

**Figure 16-10. Examples of Multiple Interrupt Servicing (2/2)**

**Example 3. Multiple interrupt servicing does not occur because interrupts are not enabled**




Interrupts are not enabled during servicing of interrupt INTxx (EI instruction is not issued), therefore, interrupt request INTyy is not acknowledged and multiple interrupt servicing does not take place. The INTyy interrupt request is held pending, and is acknowledged following execution of one main processing instruction.

PR = 00: Specify level 0 with ××PR1× = 0, ××PR0× = 0 (higher priority level)
PR = 01: Specify level 1 with ××PR1× = 0, ××PR0× = 1
PR = 10: Specify level 2 with ××PR1× = 1, ××PR0× = 0
PR = 11: Specify level 3 with ××PR1× = 1, ××PR0× = 1 (lower priority level)
IE = 0: Interrupt request acknowledgment is disabled
IE = 1: Interrupt request acknowledgment is enabled.

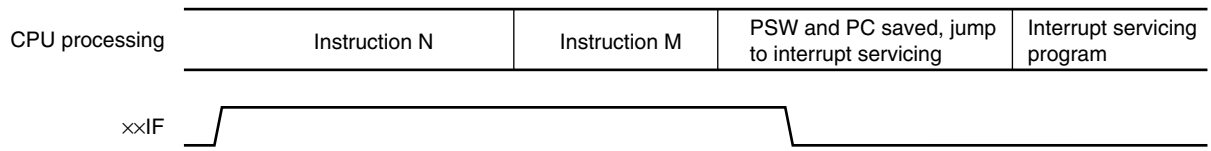
### 16.4.4 Interrupt request hold

There are instructions where, even if an interrupt request is issued while the instructions are being executed, interrupt request acknowledgment is held pending until the end of execution of the next instruction. These instructions (interrupt request hold instructions) are listed below.

- MOV PSW, #byte
- MOV PSW, A
- MOV1 PSW. bit, CY
- SET1 PSW. bit
- CLR1 PSW. bit
- RETB
- RETI
- POP PSW
- BTCLR PSW. bit, $addr20
- EI
- DI
- SKC
- SKNC
- SKZ
- SKNZ
- SKH
- SKNH
- <R> Write instructions for the IF0L, IF0H, IF1L, IF1H, IF2L, IF2H, MK0L, MK0H, MK1L, MK1H, MK2L, MK2H, PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR10L, PR10H, PR11L, PR11H, PR12L, and PR12H registers

Figure 16-11 shows the timing at which interrupt requests are held pending.

**Figure 16-11. Interrupt Request Hold**




**Remarks 1.** Instruction N: Interrupt request hold instruction

**2.** Instruction M: Instruction other than interrupt request hold instruction

# CHAPTER 17 KEY INTERRUPT FUNCTION

The number of key interrupt input channels differs, depending on the product.

| | 25-pin | 32-pin | 48-pin | 64-pin |
|---|---|---|---|---|
| Key interrupt input channels | 0 (4) ch | 1 (6) ch | 6 ch | 10 ch |

**Remarks 1.** Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR).

**2.** Most of the following descriptions in this chapter use the 64-pin products

## 17.1 Functions of Key Interrupt

A key interrupt (INTKR) can be generated by inputting a rising/falling edge to the key interrupt input pins (KR0 to KR9).

There are two ways to identify the channel(s) to which a valid edge has been input:

- Identify the channel(s) (KR0 to KR9) by using the port input level.
- Identify the channel(s) (KR0 to KR5) by using the key interrupt flag.

**Table 17-1. Assignment of Key Interrupt Detection Pins**

| Key Interrupt Pins | Key Return Mode Registers (KRM0, KRM1) | Key Return Flag Register (KRF) |
|---|---|---|
| KR0 | KRM00 | KRF0 |
| KR1 | KRM01 | KRF1 |
| KR2 | KRM02 | KRF2 |
| KR3 | KRM03 | KRF3 |
| KR4 | KRM04 | KRF4 |
| KR5 | KRM05 | KRF5 |
| KR6 | KRM06 | – |
| KR7 | KRM07 | – |
| KR8 | KRM08 | – |
| KR9 | KRM09 | – |

**Remark** (KR0 to KR3): 25-pin products
KR0 (KR0 to KR5): 32-pin products
KR0 to KR5: 48-pin products
KR0 to KR9: 64-pin products
Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR)

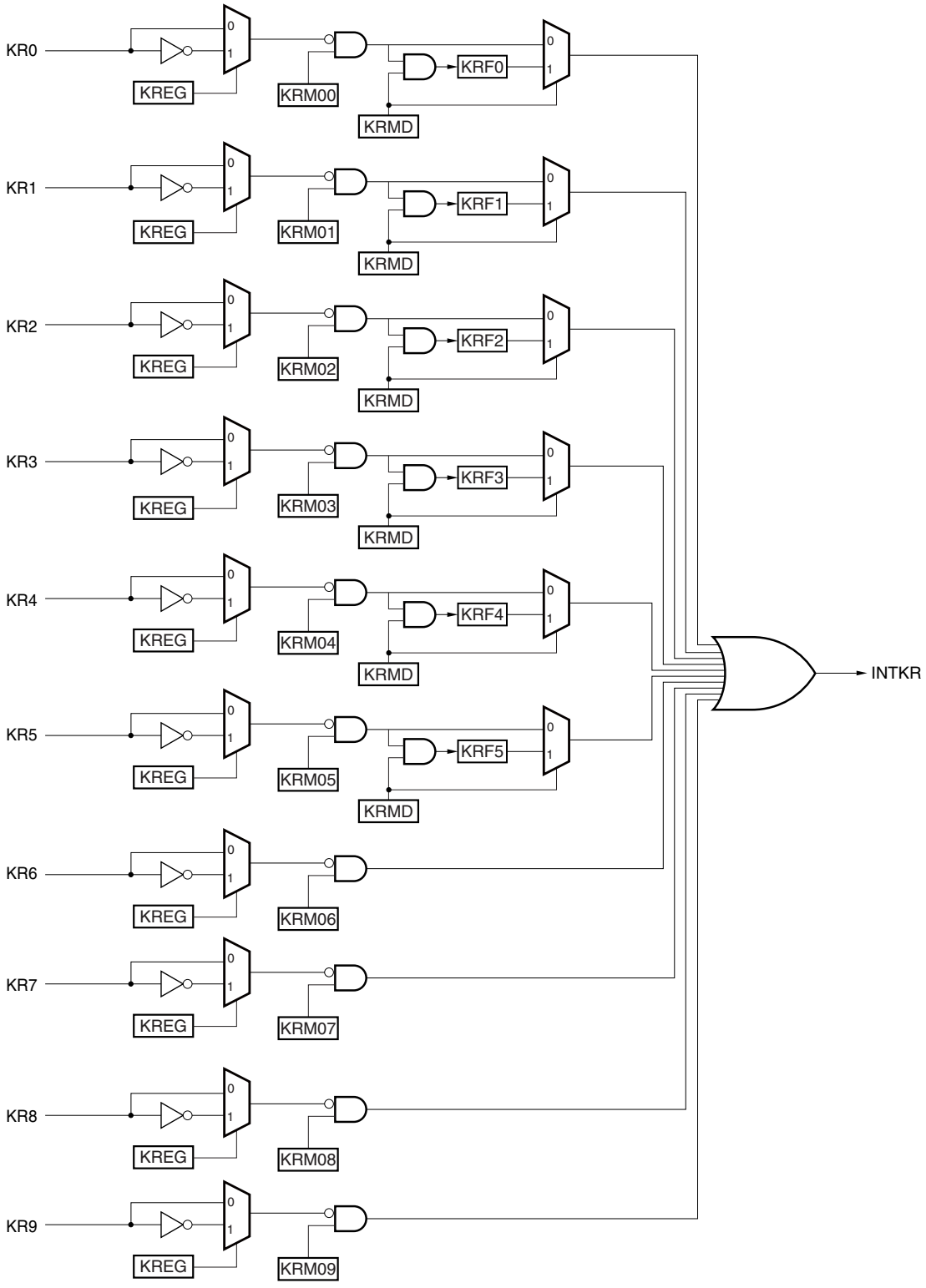
## 17.2 Configuration of Key Interrupt

The key interrupt includes the following hardware.

**Table 17-2. Configuration of Key Interrupt**

| Item | Configuration |
|---|---|
| Control register | Key return control register (KRCTL)<br>Key return mode registers 0, 1 (KRM0, KRM1)<br>Key return flag register (KRF)<br>Port mode registers 0 to 2, 7, 12, 15 (PM0 to PM2, PM7, PM12, PM15)<br>Peripheral I/O redirection register (PIOR) |

**Figure 17-1. Block Diagram of Key Interrupt**




**Remark** (KR0 to KR3): 25-pin products
KR0 (KR0 to KR5): 32-pin products
KR0 to KR5: 48-pin products
KR0 to KR9: 64-pin products
Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR)

## 17.3 Register Controlling Key Interrupt

The key interrupt function is controlled by the following five registers:

- Key return control register (KRCTL)
- Key return mode registers 0, 1 (KRM0, KRM1)
- Key return flag register (KRF)
- Port mode registers 0 to 2, 7, 12, and 15 (PM0 to PM2, PM7, PM12, PM15)
- Peripheral I/O redirection register (PIOR)

### 17.3.1 Key return control register (KRCTL)

This register controls the usage of the key interrupt flags (KRF0 to KRF5) and sets the detection edge.

The KRCTL register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

**Figure 17-2. Format of Key Return Control Register (KRCTL)**

Address: FFF34H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| KRCTL | KRMD | 0 | 0 | 0 | 0 | 0 | 0 | KREG |

| KRMD | Usage of t key interrupt flags (KRF0 to KRF5) |
|---|---|
| 0 | Does not use key interrupt flags |
| 1 | Uses key interrupt flags |

| KREG | Selection of detection edge (KR0 to KR9) |
|---|---|
| 0 | Falling edge |
| 1 | Rising edge |

### 17.3.2 Key return mode registers 0, 1 (KRM0, KRM1)

These registers set the key interrupt mode.

The KRM0 and KRM1 registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets these registers to 00H.

**Figure 17-3. Format of Key Return Mode Registers 0, 1 (KRM0, KRM1)**

Address: FFF37H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| KRM0 | KRM07 | KRM06 | KRM05 | KRM04 | KRM03 | KRM02 | KRM01 | KRM00 |

Address: FFF36H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| KRM1 | 0 | 0 | 0 | 0 | 0 | 0 | KRM09 | KRM08 |

| KRM0n | Key interrupt mode control |
|---|---|
| 0 | Does not detect key interrupt signal |
| 1 | Detects key interrupt signal |

**Cautions 1. The on-chip pull-up resistors can be applied by setting the corresponding key interrupt input pins (bits) in pull-up resistor registers 0, 1, 7, and 12 (PU0, PU1, PU7, PU12) to 1.**

**2. An interrupt will be generated if the target bit of the KRM0, KRM1 registers are set while a low level (when KREG = 0)/high level (when KREG = 1) is being input to the key interrupt input pin.**
**To ignore this interrupt, set the KRM0, KRM1 registers after disabling interrupt servicing by using the interrupt mask flag. Afterward, clear the interrupt request flag and enable interrupt servicing after waiting for the key interrupt input high-level width/low-level width (see 29.4 AC Characteristics and 30.4 AC Characteristics).**

**3. The pins not used in the key interrupt mode can be used as normal ports.**

**Remarks 1.** n = 0 to 9

**2.** (KR0 to KR3): 25-pin products
KR0 (KR0 to KR5): 32-pin products
KR0 to KR5: 48-pin products
KR0 to KR9: 64-pin products
Functions in parentheses in the above figure can be assigned via settings in the peripheral I/O redirection register (PIOR)

### 17.3.3 Key return flag register (KRF)

This register controls the key interrupt flags (KRF0 to KRF5).

The KRF register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 17-4. Format of Key return Flag Register (KRF)**

Address: FFF35H After reset: 00H R/W[Note]

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| KRF | 0 | 0 | KRF5 | KRF4 | KRF3 | KRF2 | KRF1 | KRF0 |

| KRFn | Key interrupt flag |
|---|---|
| 0 | No key interrupt signal has been detected. |
| 1 | A key interrupt signal has been detected. |

**Note** Writing to 1 is invalid. To clear KRFn, write “0” to the target bits and write “1” to other bits, with the 8-bit memory manipulation instruction.

**Caution For KR6 to KR9, identify channels by sequentially verifying the input levels.**

**Remark** n = 0 to 5

### 17.3.4 Port mode registers 0 to 2, 7, 12, 15 (PM0 to PM2, PM7, PM12, PM15)

<R> When using P70/KR0 to P77/KR7, P05/KR8, P06/KR9, P00/(KR0) to P04/(KR4), P22/(KR5) to P26/(KR9), P10/(KR0) to P15/(KR5), P120/(KR0), P151/(KR6) to P154/(KR9) as a key input, set the bits corresponding to port mode registers 0 to 2, 7, 12, 15 (PM0 to PM2, PM7, PM12, PM15) to 1. At this time, the output latch of the bits corresponding to port registers 0 to 2, 7, 12, 15 (P0 to P2, P7, P12, P15) may be 0 or 1.

The PM0 to PM2, PM7, PM12, PM15 registers can be set by a 1-bit or 8-bit memory manipulation instruction. Reset signal generation sets these registers to FFH.

In addition, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option registers 0, 1, 7, 12 (PU0, PU1, PU7, PU12).

**Remark** Functions in parentheses can be assigned via settings in the peripheral I/O redirection register (PIOR)

**Figure 17-5. Format of Port Mode Register 0 to 2, 7, 12, 15 (PM0 to PM2, PM7, PM12, PM15)**

Address: FFF20H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM0 | 1 | PM06 | PM05 | PM04 | PM03 | PM02 | PM01 | PM00 |

Address: FFF21H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM1 | 1 | PM16 | PM15 | PM14 | PM13 | PM12 | PM11 | PM10 |

Address: FFF22H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM2 | PM27 | PM26 | PM25 | PM24 | PM23 | PM22 | PM21 | PM20 |

Address: FFF27H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM7 | PM77 | PM76 | PM75 | PM74 | PM73 | PM72 | PM71 | PM70 |

Address: FFF2CH After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM12 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | PM120 |

Address: FFF2FH After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PM15 | 1 | 1 | 1 | PM154 | PM153 | PM152 | PM151 | PM150 |

| PMmn | I/O mode selection for Pmn/KRk pin (m = 0 to 2, 7, 12, 15, n = 0 to 7, k = 0 to 9) |
|---|---|
| 0 | Output mode (output buffer on) |
| 1 | Input mode (output buffer off) |

### 17.3.5 Peripheral I/O redirection register (PIOR)

This register is used to specify whether to enable or disable the peripheral I/O redirect function.

This function is used to switch ports to which alternate functions are assigned.

Use the PIOR register to assign a port to the function to redirect and enable the function.

In addition, can be changed the settings for redirection until its function enable operation. The PIOR register can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

**Figure 17-6. Format of Peripheral I/O Redirection Register (PIOR)**

Address: F0077H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| PIOR | 0 | 0 | 0 | 0 | 0 | 0 | PIOR1 | PIOR0 |

| Function | 64-pin | | | | 48-pin | | | | 32-pin | | | | 25-pin | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Setting value of PIOR1, PIOR0 | | | | Setting value of PIOR1, PIOR0 | | | | Setting value of PIOR1, PIOR0 | | | | Setting value of PIOR1, PIOR0 | | | |
| | 0, 0 | 0, 1 | 1, 0 | 1, 1 | 0, 0 | 0, 1 | 1, 0 | 1, 1 | 0, 0 | 0, 1 | 1, 0 | 1, 1 | 0, 0 | 0, 1 | 1, 0 | 1, 1 |
| KR0 | P70 | Setting prohibited | P00 | P10 | P70 | Setting prohibited | P02 | P10 | P70 | Setting prohibited | P120 | P10 | – | Setting prohibited | P02 | Setting prohibited |
| KR1 | P71 | | P01 | P11 | P71 | | P03 | P11 | – | | P02 | P11 | – | | P03 | |
| KR2 | P72 | | P02 | P12 | P72 | | P22 | P12 | – | | P03 | P12 | – | | P22 | |
| KR3 | P73 | | P03 | P13 | P73 | | P23 | P13 | – | | P22 | P13 | – | | P23 | |
| KR4 | P74 | | P04 | P14 | P74 | | P24 | P14 | – | | P23 | P14 | – | | – | |
| KR5 | P75 | | P22 | P15 | P75 | | P25 | P15 | – | | P24 | P15 | – | | – | |
| KR6 | P76 | | P23 | P151 | – | | – | – | – | | – | – | – | | – | |
| KR7 | P77 | | P24 | P152 | – | | – | – | – | | – | – | – | | – | |
| KR8 | P05 | | P25 | P153 | – | | – | – | – | | – | – | – | | – | |
| KR9 | P06 | | P26 | P154 | – | | – | – | – | | – | – | – | | – | |

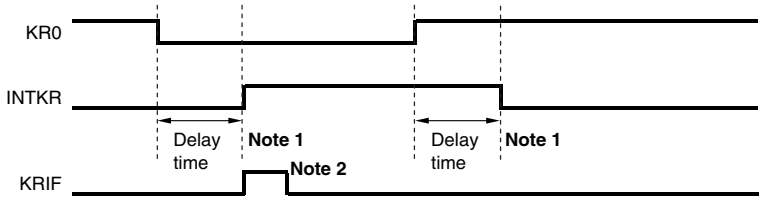
## 17.4 Key Interrupt Operation

### 17.4.1 When not using the key interrupt flag (KRMD = 0)

A key interrupt (INTKR) is generated when the valid edge specified by the setting of the KREG bit is input to a key interrupt pin (KR0 to KR9). The channel to which the valid edge was input can be identified by reading the port register and checking the port level after the key interrupt (INTKR) is generated.

The INTKR signal changes according to the input level of the key interrupt input pin (KR0 to KR9).

**Figure 17-7. Operation of INTKR Signal When a Key Interrupt is Input to a Single Channel (When KRMD = 0 and KREG = 0)**

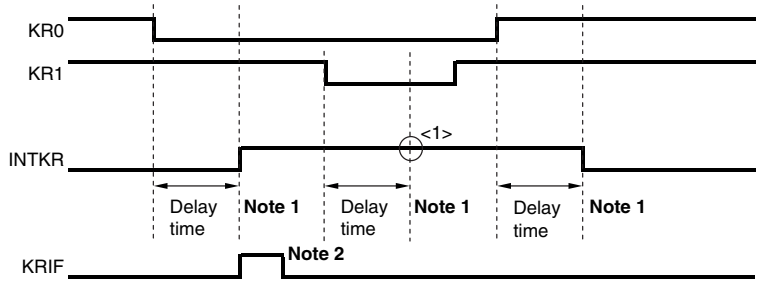



**Notes 1.** The maximum delay time is the maximum value of the high-level width and low-level width of the key interrupt input (see **29.4 AC Characteristics** and **30.4 AC Characteristics** for details).

**2.** Acknowledgment of vectored interrupt request or bit cleared by software

The operation when a valid edge is input to multiple key interrupt input pins is shown in Figure 17-8 below. The INTKR signal is set while a low level is being input to one pin (when KREG is set to 0). Therefore, even if a falling edge is input to another pin in this period, a key interrupt (INTKR) will not be generated again (<1> in the figure).

**Figure 17-8. Operation of INTKR Signal When Key Interrupts Are Input to Multiple Channels (When KRMD = 0 and KREG = 0)**




**Notes 1.** The maximum delay time is the maximum value of the high-level width and low-level width of the key interrupt input (see **29.4 AC Characteristics** and **30.4 AC Characteristics** for details).

**2.** Acknowledgment of vectored interrupt request or bit cleared by software

### 17.4.2 When using the key interrupt flag (KRMD = 1)

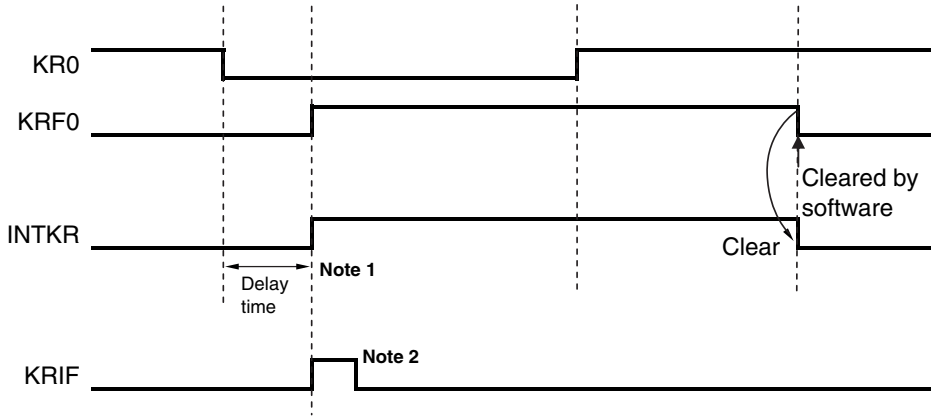
A key interrupt (INTKR) is generated when the valid edge specified by the setting of the KREG bit is input to a key interrupt pin (KR0 to KR5). The channels to which the valid edge was input can be identified by reading the key return flag register (KRF) after the key interrupt (INTKR) is generated.

If the KRMD bit is set to 1, the INTKR signal is cleared by clearing the corresponding bit in the KRF register.

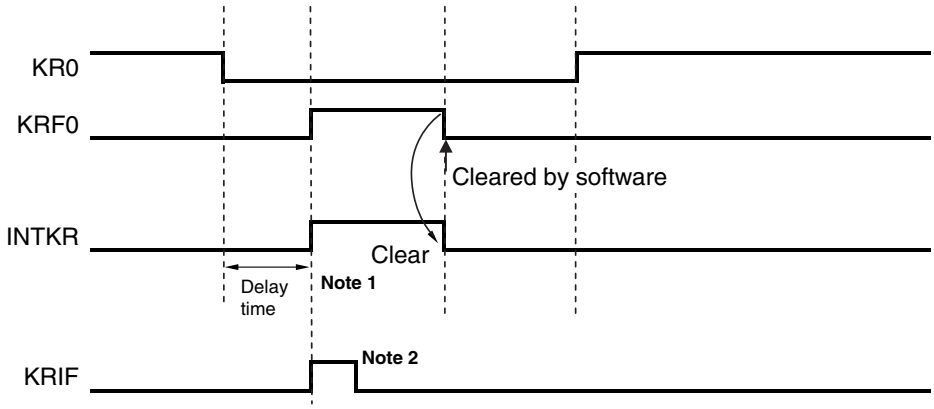
As shown in Figure 17-9, only one interrupt is generated each time a falling edge is input to one channel (when KREG = 0), regardless of whether the KRFn bit is cleared before or after a rising edge is input.

**Figure 17-9. Basic Operation of the INTKR Signal When the Key Interrupt Flag Is Used (When KRMD = 1 and KREG = 0)**

(a) When KRF0 is cleared after a rising edge is input to the KR0 pin




(b) When KRF0 is cleared before a rising edge is input to the KR0 pin

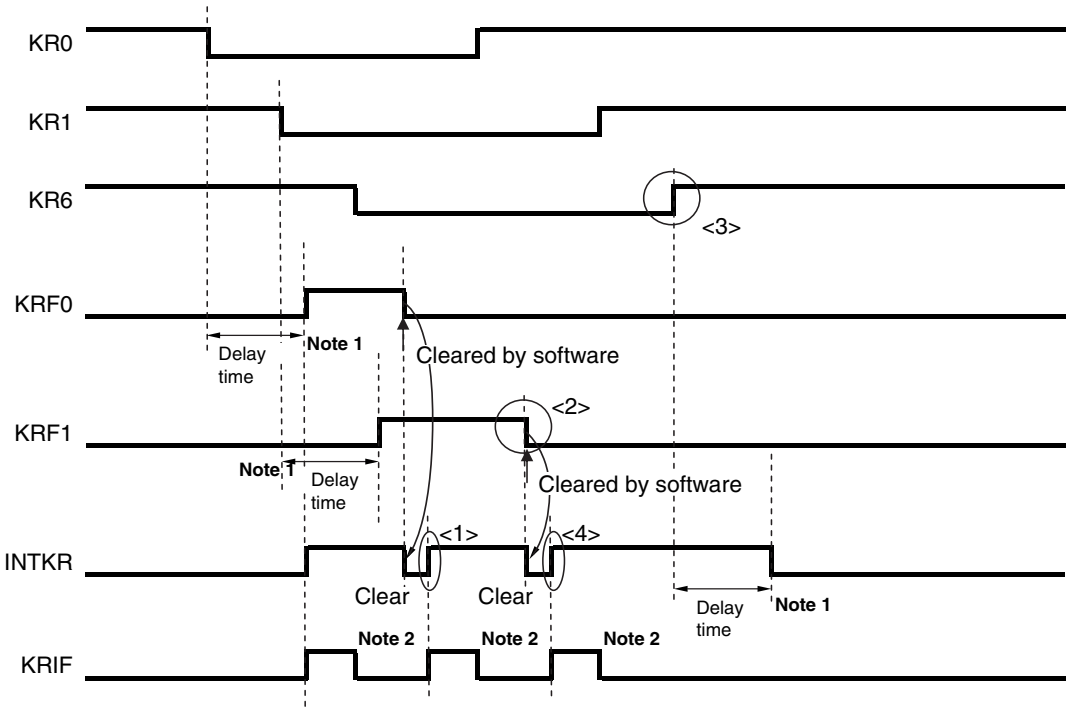



**Notes 1.** The maximum delay time is the maximum value of the high-level width and low-level width of the key interrupt input (see **29.4 AC Characteristics** and **30.4 AC Characteristics** for details).

**2.** Acknowledgment of vectored interrupt request or bit cleared by software

The operation when a valid edge is input to multiple key interrupt input pins is shown in Figure 17-10 below. A falling edge is also input to the KR1 and KR6 pins after a falling edge was input to the KR0 pin (when KREG = 0). The KRF1 bit is set when the KRF0 bit is cleared. A key interrupt (INTKR) is therefore generated one clock ($f_{CLK}$) after the KRF0 bit is cleared (<1> in the figure). Also, after a falling edge has been input to the KR6 pin, a low level continues to be input to this pin (<3> in the figure) until the KRF1 bit is cleared (<2> in the figure). A key interrupt (INTKR) is therefore generated one clock ($f_{CLK}$) after the KRF1 bit is cleared (<4> in the figure). It is thus possible to generate a key interrupt (INTKR) when a valid edge is input to multiple channels.

**Figure 17-10. Operation of INTKR Signal When Key Interrupts Are Input to Multiple Channels (When KRMD = 1 and KREG = 0)**



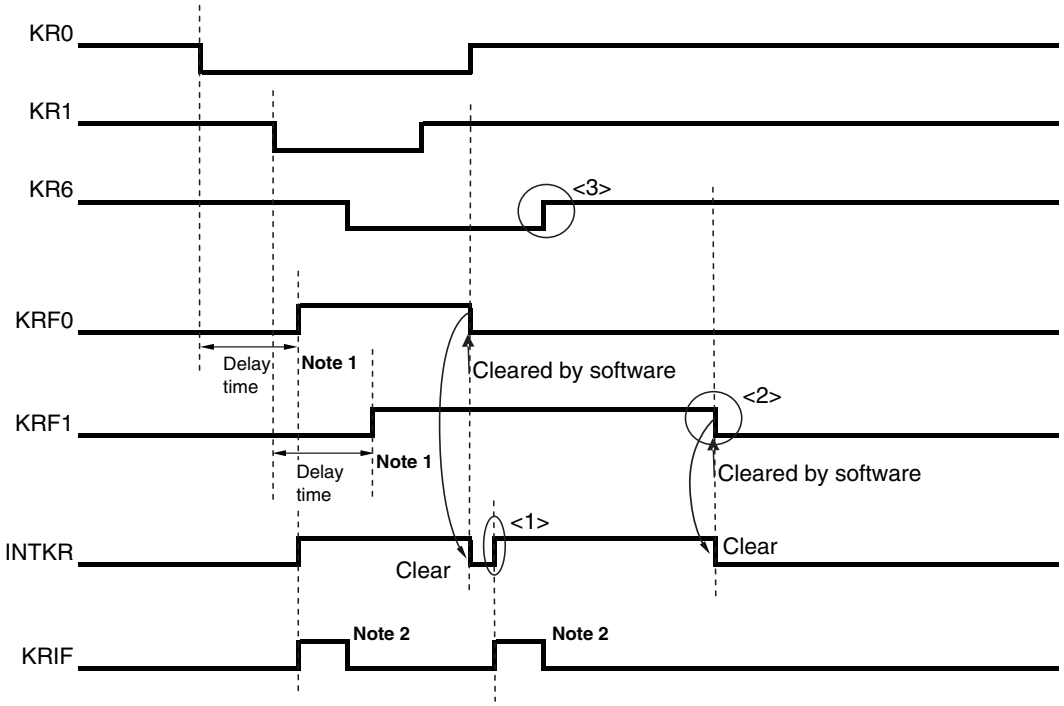

**Notes 1.** The maximum delay time is the maximum value of the high-level width and low-level width of the key interrupt input (see **29.4 AC Characteristics** and **30.4 AC Characteristics** for details).

**2.** Acknowledgment of vectored interrupt request or bit cleared by software

**Remark** $f_{CLK}$: CPU/peripheral hardware clock frequency

The operation when a valid edge is input to the KR6 to KR9 pins without generating a key interrupt (INTKR) is shown in Figure 17-11 below. A falling edge is also input to the KR1 and KR6 pins after a falling edge was input to the KR0 pin (when KREG = 0). The KR1 pin becomes high level when the KRF0 bit is cleared, but because the KRF1 bit is set, a key interrupt (INTKR) is generated one clock ($f_{CLK}$) after the KRF0 bit is cleared (<1> in the figure). Also, because the KR6 pin was high level (<3> in the figure) before the KRF1 bit was cleared (<2> in the figure) a key interrupt (INTKR) is not generated for the KR6 pin.

**Figure 17-11. Operation When an INTKR Signal Is Not Generated upon Input of a Valid Edge to KR6 to KR9 (When KRMD = 1 and KREG = 0)**




**Notes 1.** The maximum delay time is the maximum value of the high-level width and low-level width of the key interrupt input (see **29.4 AC Characteristics** and **30.4 AC Characteristics** for details).

**2.** Acknowledgment of vectored interrupt request or bit cleared by software

**Remark** $f_{CLK}$: CPU/peripheral hardware clock frequency

# CHAPTER 18 STANDBY FUNCTION

## 18.1 Standby Function

The standby function reduces the operating current of the system, and the following three modes are available.

**(1) HALT mode**

HALT instruction execution sets the HALT mode. In the HALT mode, the CPU operation clock is stopped. If the high-speed system clock oscillator, high-speed on-chip oscillator, or subsystem clock oscillator is operating before the HALT mode is set, oscillation of each clock continues. In this mode, the operating current is not decreased as much as in the STOP mode, but the HALT mode is effective for restarting operation immediately upon interrupt request generation and carrying out intermittent operations frequently.

**(2) STOP mode**

STOP instruction execution sets the STOP mode. In the STOP mode, the high-speed system clock oscillator and high-speed on-chip oscillator stop, stopping the whole system, thereby considerably reducing the CPU operating current.

Because this mode can be cleared by an interrupt request, it enables intermittent operations to be carried out. However, because a wait time is required to secure the oscillation stabilization time after the STOP mode is released when the X1 clock is selected, select the HALT mode if it is necessary to start processing immediately upon interrupt request generation.

**(3) SNOOZE mode**

In the case of CSIp or UARTq data reception and an A/D conversion request by the timer trigger signal (the interrupt request signal (INTRTC/INTIT)), the STOP mode is exited, the CSIp or UARTq data is received without operating the CPU, and A/D conversion is performed. This can only be specified when the high-speed on-chip oscillator is selected for the CPU/peripheral hardware clock ($f_{CLK}$).

In either of these two modes, all the contents of registers, flags and data memory just before the standby mode is set are held. The I/O port output latches and output buffer statuses are also held.

**Cautions 1. The STOP mode can be used only when the CPU is operating on the main system clock. Do not set to the STOP mode while the CPU operates with the subsystem clock. The HALT mode can be used when the CPU is operating on either the main system clock or the subsystem clock.**

**2. When shifting to the STOP mode, be sure to stop the peripheral hardware operation operating with main system clock before executing STOP instruction (except SNOOZE mode setting unit).**

**3. When using CSIp, UARTq, or the A/D converter in the SNOOZE mode, set up serial standby control register m (SSCm) and A/D converter mode register 2 (ADM2) before switching to the STOP mode. For details, see 12.3 Registers Controlling Serial Array Unit and 11.3 Registers Used in A/D Converter.**

**4. The following sequence is recommended for power consumption reduction of the A/D converter when the standby function is used: First clear bit 7 (ADCS) and bit 0 (ADCE) of A/D converter mode register 0 (ADM0) to 0 to stop the A/D conversion operation, and then execute the STOP instruction.**

**5. It can be selected by the option byte whether the low-speed on-chip oscillator continues oscillating or stops in the HALT or STOP mode. For details, see CHAPTER 24 OPTION BYTE.**

**Remark** p = 00; q = 0; m = 0

## 18.2 Registers Controlling Standby Function

The registers which control the standby function are described below.

<R>
- Subsystem clock supply mode control register (OSMC)
- Oscillation stabilization time counter status register (OSTC)
- Oscillation stabilization time select register (OSTS)

**Remark** For details of registers described above, see **CHAPTER 5 CLOCK GENERATOR**. For registers which control the SNOOZE mode, **CHAPTER 11 A/D CONVERTER** and **CHAPTER 12 SERIAL ARRAY UNIT**.

## 18.3 Standby Function Operation

### 18.3.1 HALT mode

**(1) HALT mode**

The HALT mode is set by executing the HALT instruction. HALT mode can be set regardless of whether the CPU clock before the setting was the high-speed system clock, high-speed on-chip oscillator clock, or subsystem clock. The operating statuses in the HALT mode are shown below.

<R> **Caution Because the interrupt request signal is used to clear the HALT mode, if the interrupt mask flag is 0 (the interrupt processing is enabled) and the interrupt request flag is 1 (the interrupt request signal is generated), the HALT mode is not entered even if the HALT instruction is executed in such a situation.**

**Table 18-1. Operating Statuses in HALT Mode (1/2)**

| HALT Mode Setting / Item | | | When HALT Instruction Is Executed While CPU Is Operating on Main System Clock | | |
|---|---|---|---|---|---|
| | | | When CPU Is Operating on High-speed On-chip Oscillator Clock ($f_{IH}$) | When CPU Is Operating on X1 Clock ($f_X$) | When CPU Is Operating on External Main System Clock ($f_{EX}$) |
| System clock | | | Clock supply to the CPU is stopped | | |
| | Main system clock | $f_{IH}$ | Operation continues (cannot be stopped) | Operation disabled | |
| | | $f_X$ | Operation disabled | Operation continues (cannot be stopped) | Cannot operate |
| | | $f_{EX}$ | | Cannot operate | Operation continues (cannot be stopped) |
| | Subsystem clock | $f_{XT}$ | Status before HALT mode was set is retained | | |
| | | $f_{EXS}$ | | | |
| | $f_{IL}$ | | Set by bits 0 (WDSTBYON) and 4 (WDTON) of option byte (000C0H), and WUTMMCK0 bit of subsystem clock supply mode control register (OSMC)<br>• WUTMMCK0 = 1: Oscillates<br>• WUTMMCK0 = 0 and WDTON = 0: Stops<br>• WUTMMCK0 = 0, WDTON = 1, and WDSTBYON = 1: Oscillates<br>• WUTMMCK0 = 0, WDTON = 1, and WDSTBYON = 0: Stops | | |
| CPU | | | Operation stopped | | |
| Code flash memory | | | | | |
| Data flash memory | | | | | |
| RAM | | | | | |
| Port (latch) | | | Status before HALT mode was set is retained | | |
| Timer array unit | | | Operable | | |
| Real-time clock (RTC) | | | | | |
| 12-bit interval timer | | | | | |
| Watchdog timer | | | See **CHAPTER 10 WATCHDOG TIMER** | | |
| Clock output/buzzer output | | | Operable | | |
| A/D converter | | | | | |
| Serial array unit (SAU) | | | | | |
| Serial interface (IICA) | | | | | |
| Multiplier and divider/multiply-accumulator | | | | | |
| DMA controller | | | | | |
| Power-on-reset function | | | | | |
| Voltage detection function | | | | | |
| External interrupt | | | | | |
| Key interrupt function | | | | | |
| CRC operation function | High-speed CRC | | | | |
| | General-purpose CRC | | Operation stopped | | |
| RAM parity error detection function | | | Operation stopped | | |
| RAM guard function | | | | | |
| SFR guard function | | | | | |
| Illegal-memory access detection function | | | | | |

**Remark** Operation stopped: Operation is automatically stopped before switching to the HALT mode.
Operation disabled: Operation is stopped before switching to the HALT mode.
$f_{IH}$: High-speed on-chip oscillator clock $f_{EX}$: External main system clock
$f_{IL}$: Low-speed on-chip oscillator clock $f_{XT}$: XT1 clock
$f_X$: X1 clock $f_{EXS}$: External subsystem clock

**Table 18-1. Operating Statuses in HALT Mode (2/2)**

| Item \ HALT Mode Setting | | When HALT Instruction Is Executed While CPU Is Operating on Subsystem Clock | |
|---|---|---|---|
| | | When CPU Is Operating on XT1 Clock ($f_{XT}$) | When CPU Is Operating on External Subsystem Clock ($f_{EXS}$) |
| System clock | | Clock supply to the CPU is stopped | |
| Main system clock | $f_{IH}$ | Operation disabled | |
| | $f_X$ | | |
| | $f_{EX}$ | | |
| Subsystem clock | $f_{XT}$ | Operation continues (cannot be stopped) | Cannot operate |
| | $f_{EXS}$ | Cannot operate | Operation continues (cannot be stopped) |
| $f_{IL}$ | | Set by bits 0 (WDSTBYON) and 4 (WDTON) of option byte (000C0H), and WUTMMCK0 bit of subsystem clock supply mode control register (OSMC)<br>• WUTMMCK0 = 1: Oscillates<br>• WUTMMCK0 = 0 and WDTON = 0: Stops<br>• WUTMMCK0 = 0, WDTON = 1, and WDSTBYON = 1: Oscillates<br>• WUTMMCK0 = 0, WDTON = 1, and WDSTBYON = 0: Stops | |
| CPU | | Operation stopped | |
| Code flash memory | | | |
| Data flash memory | | | |
| RAM | | | |
| Port (latch) | | Status before HALT mode was set is retained | |
| Timer array unit | | Operates when the RTCLPC bit is 0 (operation is disabled when the RTCLPC bit is not 0). | |
| Real-time clock (RTC) | | Operable | |
| 12-bit interval timer | | | |
| Watchdog timer | | See **CHAPTER 10 WATCHDOG TIMER** | |
| Clock output/buzzer output | | Operates when the RTCLPC bit is 0 (operation is disabled when the RTCLPC bit is not 0) | |
| A/D converter | | Operation disabled | |
| Serial array unit (SAU) | | Operates when the RTCLPC bit is 0 (operation is disabled when the RTCLPC bit is not 0). | |
| Serial interface (IICA) | | Operation disabled | |
| Multiplier and divider/multiply-accumulator | | Operates when the RTCLPC bit is 0 (operation is disabled when the RTCLPC bit is not 0). | |
| DMA controller | | | |
| Power-on-reset function | | Operable | |
| Voltage detection function | | | |
| External interrupt | | | |
| Key interrupt function | | | |
| CRC operation function | High-speed CRC | Operation disabled | |
| | General-purpose CRC | Operation stopped | |
| RAM parity error detection function | | Operation stopped | |
| RAM guard function | | | |
| SFR guard function | | | |
| Illegal-memory access detection function | | | |

**Remark** Operation stopped: Operation is automatically stopped before switching to the HALT mode.
Operation disabled: Operation is stopped before switching to the HALT mode.
$f_{IH}$: High-speed on-chip oscillator clock $f_{EX}$: External main system clock
$f_{IL}$: Low-speed on-chip oscillator clock $f_{XT}$: XT1 clock
$f_X$: X1 clock $f_{EXS}$: External subsystem clock

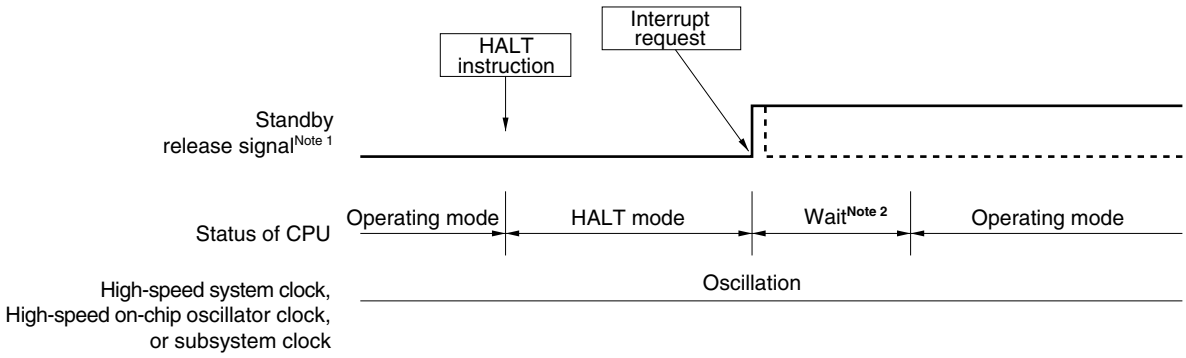
### (2) HALT mode release

The HALT mode can be released by the following two sources.

#### (a) Release by unmasked interrupt request

When an unmasked interrupt request is generated, the HALT mode is released. If interrupt acknowledgment is enabled, vectored interrupt servicing is carried out. If interrupt acknowledgment is disabled, the next address instruction is executed.

**Figure 18-1. HALT Mode Release by Interrupt Request Generation**




**Notes 1.** For details of the standby release signal, see **Figure 16-1**.

**2.** Wait time for HALT mode release

- When vectored interrupt servicing is carried out

  Main system clock: 15 to 16 clock
  Subsystem clock (RTCLPC = 0): 10 to 11 clock
  Subsystem clock (RTCLPC = 1): 11 to 12 clock

- When vectored interrupt servicing is not carried out

  Main system clock: 9 to 10 clock
  Subsystem clock (RTCLPC = 0): 4 to 5 clock
  Subsystem clock (RTCLPC = 1): 5 to 6 clock

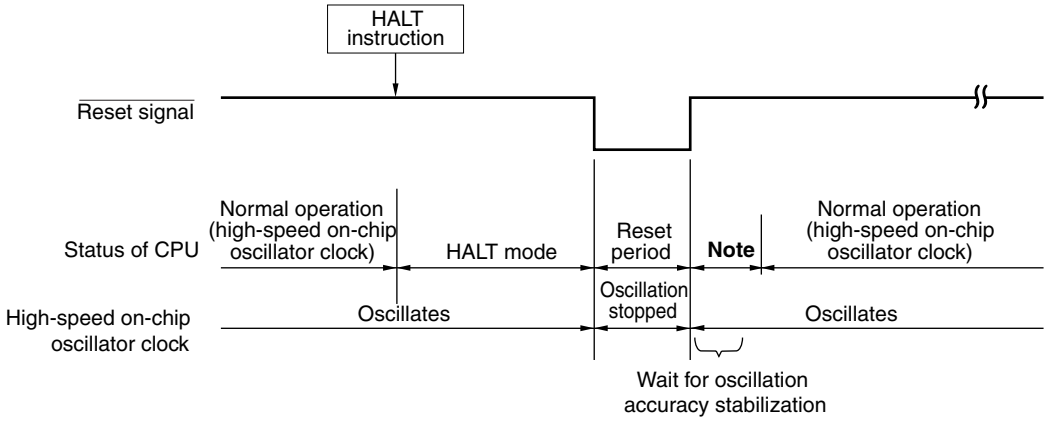
**Remark** The broken lines indicate the case when the interrupt request which has released the standby mode is acknowledged.

**(b) Release by reset signal generation**

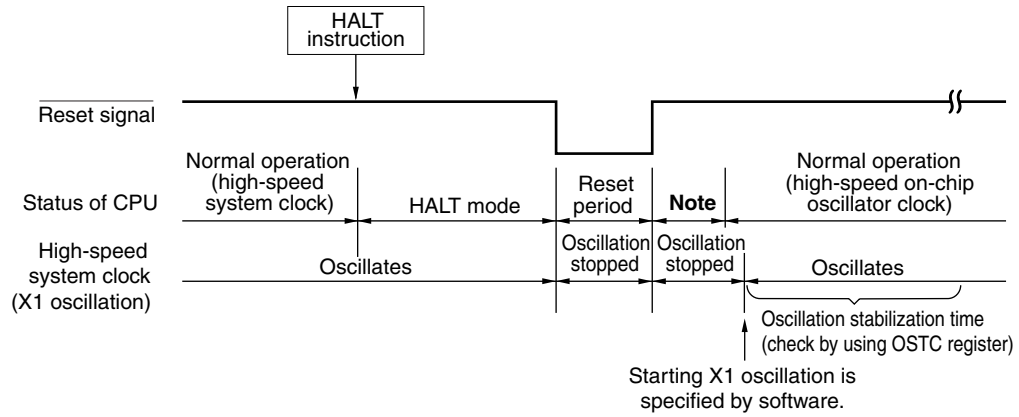
When the reset signal is generated, HALT mode is released, and then, as in the case with a normal reset operation, the program is executed after branching to the reset vector address.

**Figure 18-2. HALT Mode Release by Reset (1/2)**

<R> **(1) When high-speed on-chip oscillator clock is used as CPU clock**



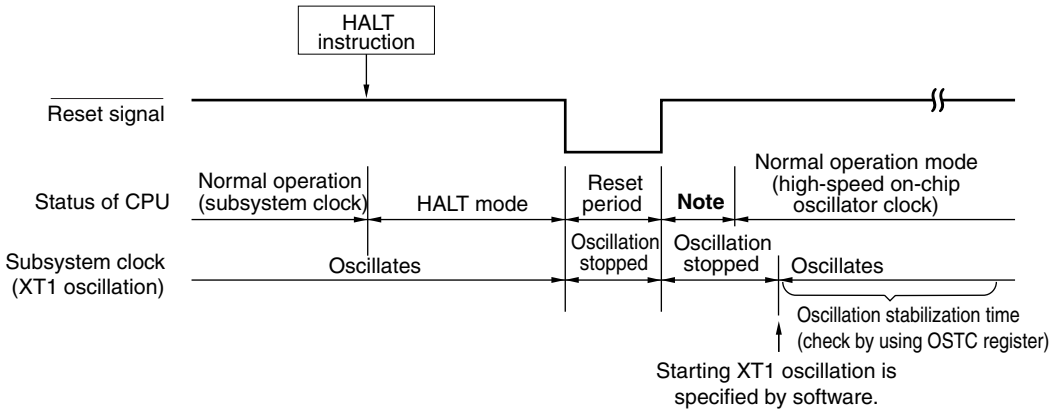

<R> **(2) When high-speed system clock is used as CPU clock**




**Note** For the reset processing time, see **CHAPTER 19 RESET FUNCTION**. For the reset processing time of the power-on-reset circuit (POR) and voltage detector (LVD), see **CHAPTER 20 POWER-ON-RESET CIRCUIT**.

**Figure 18-2. HALT Mode Release by Reset (2/2)**

**(3) When subsystem clock is used as CPU clock**




<R> **Note** For the reset processing time, see **CHAPTER 19 RESET FUNCTION**. For the reset processing time of the power-on-reset circuit (POR) and voltage detector (LVD), see **CHAPTER 20 POWER-ON-RESET CIRCUIT**.

### 18.3.2 STOP mode

**(1) STOP mode setting and operating statuses**

The STOP mode is set by executing the STOP instruction, and it can be set only when the CPU clock before the setting was the high-speed on-chip oscillator clock, X1 clock, or external main system clock.

<R> **Caution Because the interrupt request signal is used to clear the STOP mode, if the interrupt mask flag is 0 (the interrupt processing is enabled) and the interrupt request flag is 1 (the interrupt request signal is generated), the STOP mode is immediately cleared if set when the STOP instruction is executed in such a situation. Accordingly, once the STOP instruction is executed, the system returns to its normal operating mode after the elapse of release time from the STOP mode.**

**Remark** p = 00; q = 0; m = 0

The operating statuses in the STOP mode are shown below.

**Table 18-2. Operating Statuses in STOP Mode**

| STOP Mode Setting / Item | | | When STOP Instruction Is Executed While CPU Is Operating on Main System Clock | | |
|---|---|---|---|---|---|
| | | | When CPU Is Operating on High-speed on-chip oscillator clock ($f_{IH}$) | When CPU Is Operating on X1 Clock ($f_X$) | When CPU Is Operating on External Main System Clock ($f_{EX}$) |
| System clock | | | Clock supply to the CPU is stopped | | |
| | Main system clock | $f_{IH}$ | Stopped | | |
| | | $f_X$ | | | |
| | | $f_{EX}$ | | | |
| | Subsystem clock | $f_{XT}$ | Status before STOP mode was set is retained | | |
| | | $f_{EXS}$ | | | |
| | $f_{IL}$ | | Set by bits 0 (WDSTBYON) and 4 (WDTON) of option byte (000C0H), and WUTMMCK0 bit of subsystem clock supply mode control register (OSMC)<br>• WUTMMCK0 = 1: Oscillates<br>• WUTMMCK0 = 0 and WDTON = 0: Stops<br>• WUTMMCK0 = 0, WDTON = 1, and WDSTBYON = 1: Oscillates<br>• WUTMMCK0 = 0, WDTON = 1, and WDSTBYON = 0: Stops | | |
| CPU | | | Operation stopped | | |
| Code flash memory | | | | | |
| Data flash memory | | | Operation stopped | | |
| RAM | | | Operation stopped | | |
| Port (latch) | | | Status before STOP mode was set is retained | | |
| Timer array unit | | | Operation disabled | | |
| Real-time clock (RTC) | | | Operable | | |
| 12-bit interval timer | | | | | |
| Watchdog timer | | | See **CHAPTER 10 WATCHDOG TIMER** | | |
| Clock output/buzzer output | | | Operates when the subsystem clock is selected as the clock source for counting and the RTCLPC bit is 0 (operation is disabled when a clock other than the subsystem clock is selected and the RTCLPC bit is not 0). | | |
| A/D converter | | | Wakeup operation is enabled (switching to the SNOOZE mode) | | |
| Serial array unit (SAU) | | | Wakeup operation is enabled only for CSIp and UARTq (switching to the SNOOZE mode)<br>Operation is disabled for anything other than CSIp and UARTq | | |
| Serial interface (IICA) | | | Wakeup by address match operable | | |
| Multiplier and divider/multiply-accumulator | | | Operation disabled | | |
| DMA controller | | | | | |
| Power-on-reset function | | | Operable | | |
| Voltage detection function | | | | | |
| External interrupt | | | | | |
| Key interrupt function | | | | | |
| CRC operation function | High-speed CRC | | Operation stopped | | |
| | General-purpose CRC | | | | |
| RAM parity error detection function | | | | | |
| RAM guard function | | | | | |
| SFR guard function | | | | | |
| Illegal-memory access detection function | | | | | |

**Remarks 1.** Operation stopped: Operation is automatically stopped before switching to the STOP mode.
Operation disabled: Operation is stopped before switching to the STOP mode.
$f_{IH}$: High-speed on-chip oscillator clock
$f_X$: X1 clock
$f_{XT}$: XT1 clock
$f_{IL}$: Low-speed on-chip oscillator clock
$f_{EX}$: External main system clock
$f_{EXS}$: External subsystem clock

**2.** p = 00; q = 0

### (2) STOP mode release

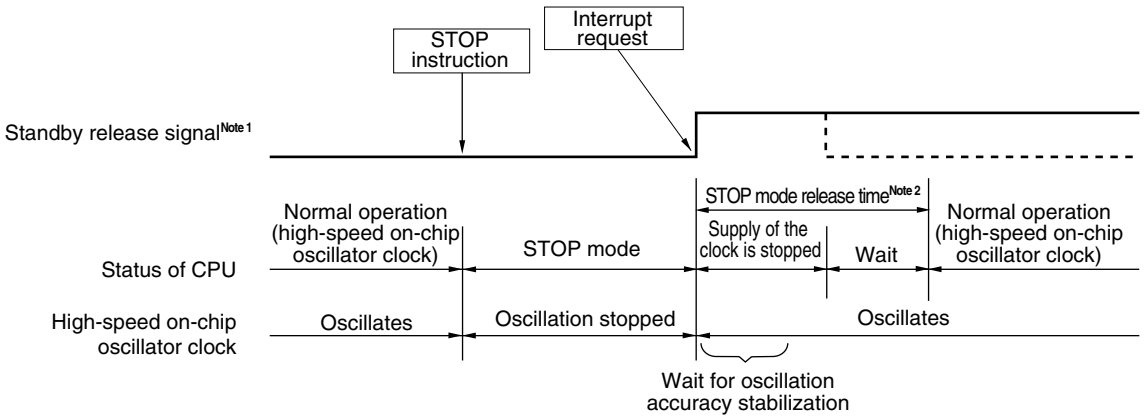
The STOP mode can be released by the following two sources.

#### (a) Release by unmasked interrupt request

When an unmasked interrupt request is generated, the STOP mode is released. After the oscillation stabilization time has elapsed, if interrupt acknowledgment is enabled, vectored interrupt servicing is carried out. If interrupt acknowledgment is disabled, the next address instruction is executed.

**Figure 18-3. STOP Mode Release by Interrupt Request Generation (1/2)**

<R> **(1) When high-speed on-chip oscillator clock is used as CPU clock**




**Notes 1.** For details of the standby release signal, see **Figure 16-1**.

**2.** STOP mode release time

Supply of the clock is stopped: 18 $\mu$s to 65 $\mu$s

Wait

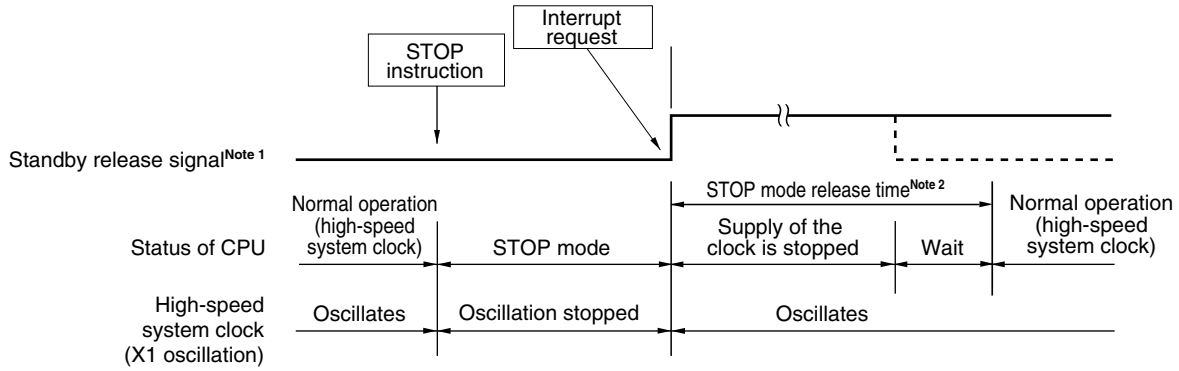
- When vectored interrupt servicing is carried out: 7 clocks
- When vectored interrupt servicing is not carried out: 1 clock

**Remarks 1.** The clock supply stop time varies depending on the temperature conditions and STOP mode period.

**2.** The broken lines indicate the case when the interrupt request that has released the standby mode is acknowledged.

**Figure 18-3. STOP Mode Release by Interrupt Request Generation (2/2)**

<R> **(2) When high-speed system clock (X1 oscillation) is used as CPU clock**




**Notes 1.** For details of the standby release signal, see **Figure 16-1**.

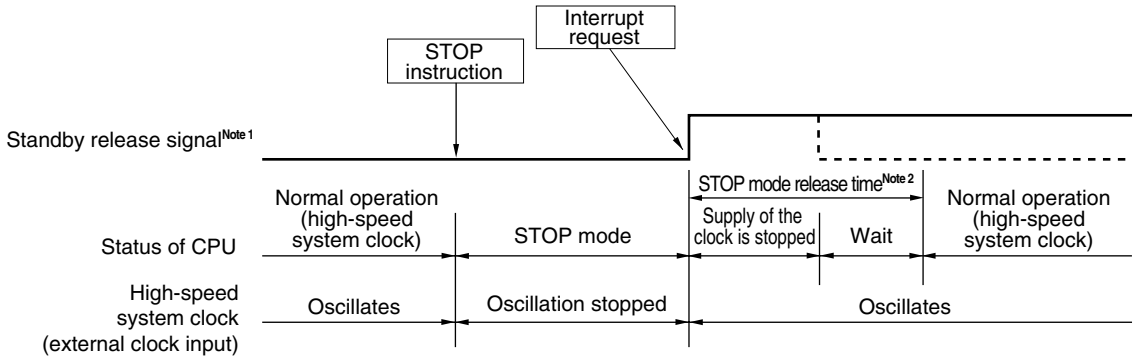
**2.** STOP mode release time

Supply of the clock is stopped: 18 $\mu$s to "whichever is longer 65 $\mu$s and the oscillation stabilization time (set by OSTS)"

Wait

- When vectored interrupt servicing is carried out: 10 to 11 clocks
- When vectored interrupt servicing is not carried out: 4 to 5 clocks

<R> **(3) When high-speed system clock (external clock input) is used as CPU clock**




<R> **Notes 1.** For details of the standby release signal, see **Figure 16-1**.

<R> **2.** STOP mode release time

Supply of the clock is stopped: 18 to 65 $\mu$s

Wait

- When vectored interrupt servicing is carried out: 7 clocks
- When vectored interrupt servicing is not carried out: 1 clock

<R> **Caution To reduce the oscillation stabilization time after release from the STOP mode while CPU operates based on the high-speed system clock (X1 oscillation), switch the clock to the high-speed on-chip oscillator clock temporarily before executing the STOP instruction.**

**Remarks 1.** The period during which clock supply stops depends on the temperature conditions and the STOP mode period.

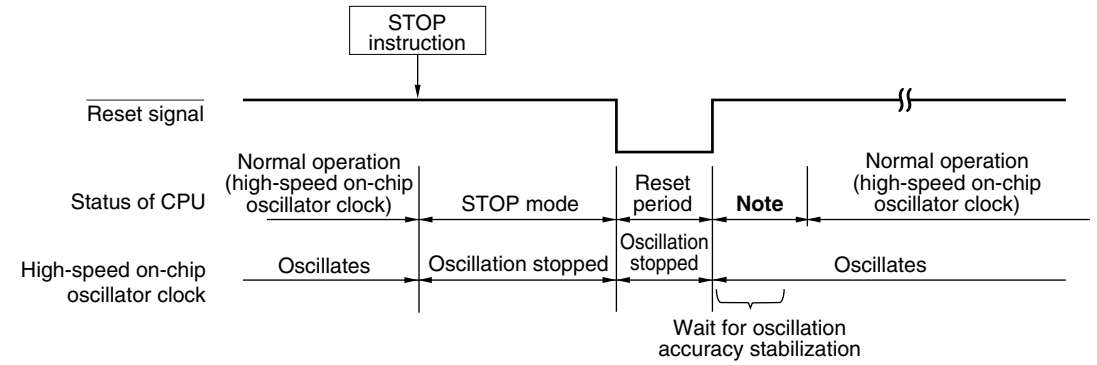
**2.** The broken lines indicate the case when the interrupt request that has released the standby mode is acknowledged.

**(b) Release by reset signal generation**

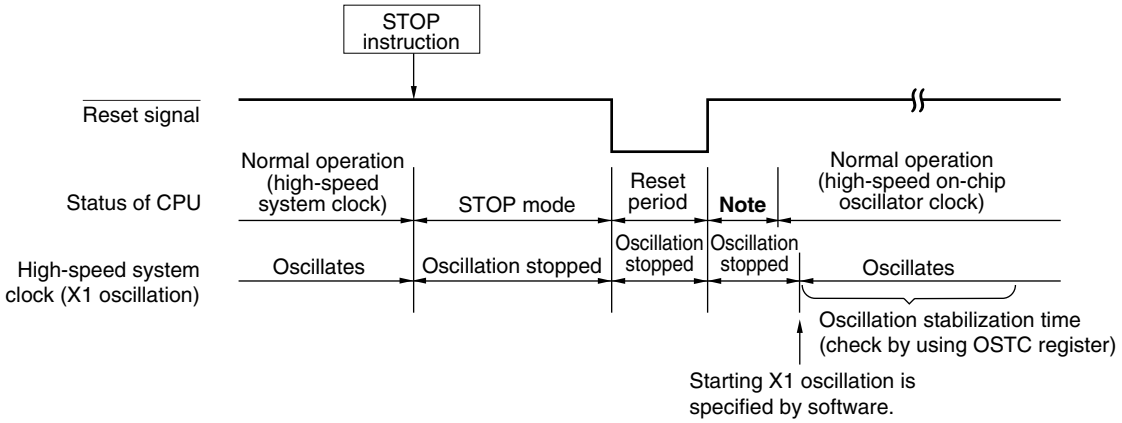
When the reset signal is generated, STOP mode is released, and then, as in the case with a normal reset operation, the program is executed after branching to the reset vector address.

**Figure 18-4. STOP Mode Release by Reset**

<R> **(1) When high-speed on-chip oscillator clock is used as CPU clock**




<R> **(2) When high-speed system clock is used as CPU clock**




<R> **Note** For the reset processing time, see **CHAPTER 19 RESET FUNCTION**. For the reset processing time of the power-on-reset circuit (POR) and voltage detector (LVD), see **CHAPTER 20 POWER-ON-RESET CIRCUIT**.

### 18.3.3 SNOOZE mode

**(1) SNOOZE mode setting and operating statuses**

The SNOOZE mode can only be specified for CSIp, UARTq, or the A/D converter. Note that this mode can only be specified if the CPU clock is the high-speed on-chip oscillator clock.

When using CSIp or UARTq in the SNOOZE mode, set the SWCm bit of the serial standby control register m (SSCm) to 1 immediately before switching to the STOP mode. For details, see **12.3 Registers Controlling Serial Array Unit**.

When using the A/D converter in the SNOOZE mode, set the AWC bit of the A/D converter mode register 2 (ADM2) to 1 immediately before switching to the STOP mode. For details, see **11.3 Registers Used in A/D Converter**.

**Remark** p = 00; q = 0; m = 0

The following time is required for mode transition.

Transition time from STOP mode to SNOOZE mode: 18 to 65 $\mu$s

**Remark** The time required to transition from STOP mode to SNOOZE mode depends on the temperature conditions and the STOP mode period.

Transition time from SNOOZE mode to normal operation:

- When vectored interrupt servicing is carried out:
  HS (High-speed main) mode: 4.99 to 9.44 $\mu$s + 7 clocks
  LS (Low-speed main) mode: 1.10 to 5.08 $\mu$s + 7 clocks
  LV (Low-voltage main) mode: 16.58 to 25.40 $\mu$s + 7 clocks
- When vectored interrupt servicing is not carried out:
  HS (High-speed main) mode: 4.99 to 9.44 $\mu$s + 1 clock
  LS (Low-speed main) mode: 1.10 to 5.08 $\mu$s + 1 clock
  LV (Low-voltage main) mode: 16.58 to 25.40 $\mu$s + 1 clock

The operating statuses in the SNOOZE mode are shown below.

**Table 18-3. Operating Statuses in SNOOZE Mode**

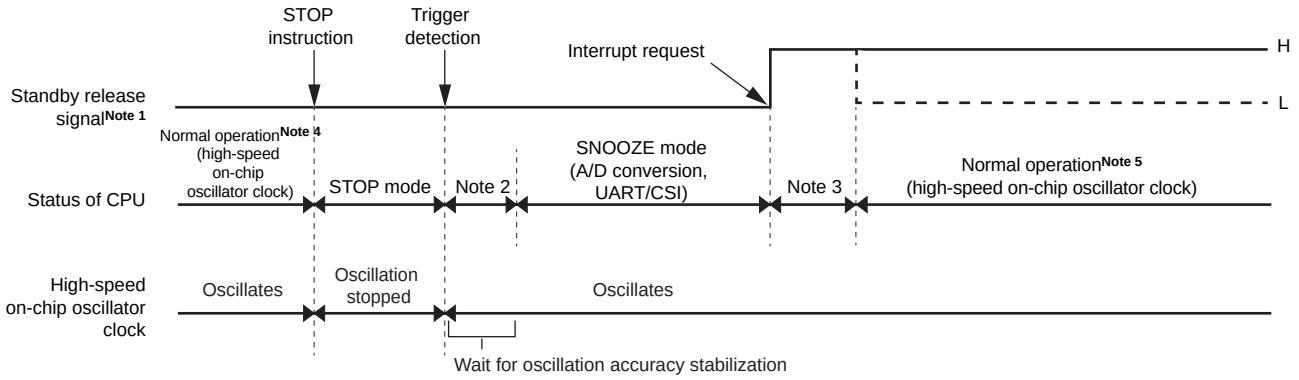
| Item \ STOP Mode Setting | | | When Inputting CSIp/UARTq Data Reception Signal or A/D Converter Timer Trigger Signal While in STOP Mode |
|---|---|---|---|
| | | | When CPU Is Operating on High-speed on-chip oscillator clock ($f_{IH}$) |
| System clock | | | Clock supply to the CPU is stopped |
| | Main system clock | $f_{IH}$ | Operation started |
| | | $f_X$ | Stopped |
| | | $f_{EX}$ | |
| | Subsystem clock | $f_{XT}$ | Use of the status while in the STOP mode continues |
| | | $f_{EXS}$ | |
| | $f_{IL}$ | | Set by bits 0 (WDSTBYON) and 4 (WDTON) of option byte (000C0H), and WUTMMCK0 bit of subsystem clock supply mode control register (OSMC)<br>• WUTMMCK0 = 1: Oscillates<br>• WUTMMCK0 = 0 and WDTON = 0: Stops<br>• WUTMMCK0 = 0, WDTON = 1, and WDSTBYON = 1: Oscillates<br>• WUTMMCK0 = 0, WDTON = 1, and WDSTBYON = 0: Stops |
| CPU | | | Operation stopped |
| Code flash memory | | | |
| Data flash memory | | | |
| RAM | | | |
| Port (latch) | | | Use of the status while in the STOP mode continues |
| Timer array unit | | | Operation disabled |
| Real-time clock (RTC) | | | Operable |
| 12-bit interval timer | | | |
| Watchdog timer | | | See **CHAPTER 10 WATCHDOG TIMER** |
| Clock output/buzzer output | | | Operates when the subsystem clock is selected as the clock source for counting and the RTCLPC bit is 0 (operation is disabled when a clock other than the subsystem clock is selected and the RTCLPC bit is not 0). |
| A/D converter | | | Operable |
| Serial array unit (SAU) | | | Operable only CSIp and UARTq only. Operation disabled other than CSIp and UARTq. |
| Serial interface (IICA) | | | Operation disabled |
| Multiplier and divider/multiply-accumulator | | | |
| DMA controller | | | |
| Power-on-reset function | | | Operable |
| Voltage detection function | | | |
| External interrupt | | | |
| Key interrupt function | | | |
| <R> CRC operation function | High-speed CRC | | Operation disabled |
| | General-purpose CRC | | |
| RAM parity error detection function | | | |
| RAM guard function | | | |
| SFR guard function | | | |
| Illegal-memory access detection function | | | |

**Remarks 1.** Operation stopped: Operation is automatically stopped before switching to the SNOOZE mode.
Operation disabled: Operation is stopped before switching to the SNOOZE mode.
$f_{IH}$: High-speed on-chip oscillator clock $f_{IL}$: Low-speed on-chip oscillator clock
$f_X$: X1 clock $f_{EX}$: External main system clock
$f_{XT}$: XT1 clock $f_{EXS}$: External subsystem clock

**2.** p = 00; q = 0

### <R> (2) Timing diagram when the interrupt request signal is generated in the SNOOZE mode

**Figure 18-5. When the Interrupt Request Signal is Generated in the SNOOZE Mode**




**Notes 1.** For details of the standby release signal, see **Figure 16-1**.

**2.** Transition time from STOP mode to SNOOZE mode

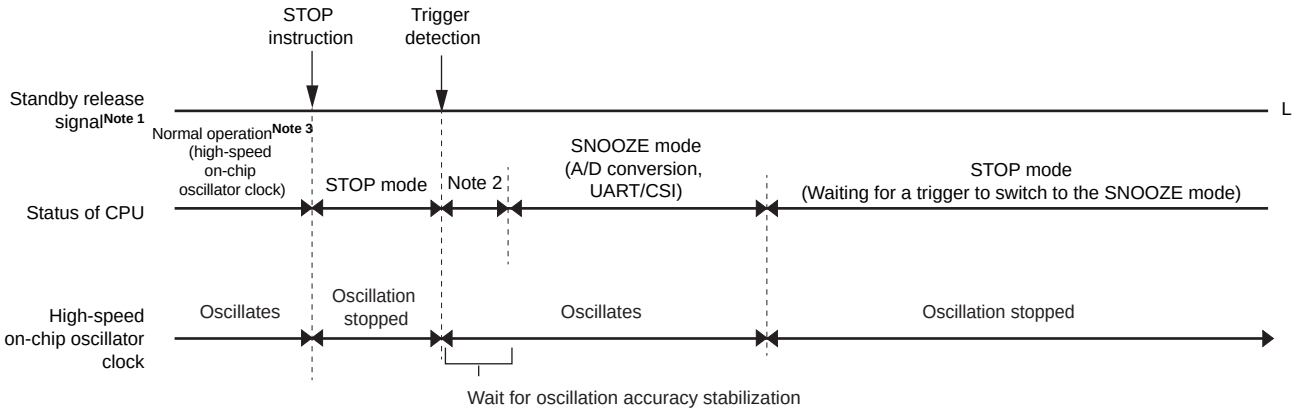
**3.** Transition time from SNOOZE mode to normal operation

**4.** Enable the SNOOZE mode (AWC = 1 or SWC = 1) immediately before switching to the STOP mode.

**5.** Be sure to release the SNOOZE mode (AWC = 0 or SWC = 0) immediately after return to the normal operation.

### <R> (3) Timing diagram when the interrupt request signal is not generated in the SNOOZE mode

**Figure 18-6. When the Interrupt Request Signal is not Generated in the SNOOZE Mode**




**Notes 1.** For details of the standby release signal, see **Figure 16-1**.

**2.** Transition time from STOP mode to SNOOZE mode

**3.** Enable the SNOOZE mode (AWC = 1 or SWC = 1) immediately before switching to the STOP mode.

**Remark** For details of the SNOOZE mode function, see **CHAPTER 11 A/D CONVERTER** and **CHAPTER 12 SERIAL ARRAY UNIT**.

# CHAPTER 19 RESET FUNCTION

The following seven operations are available to generate a reset signal.

(1) External reset input via $\overline{\text{RESET}}$ pin
(2) Internal reset by watchdog timer program loop detection
(3) Internal reset by comparison of supply voltage and detection voltage of power-on-reset (POR) circuit
(4) Internal reset by comparison of supply voltage of the voltage detector (LVD) and detection voltage
(5) Internal reset by execution of illegal instruction[Note]
(6) Internal reset by RAM parity error
(7) Internal reset by illegal-memory access

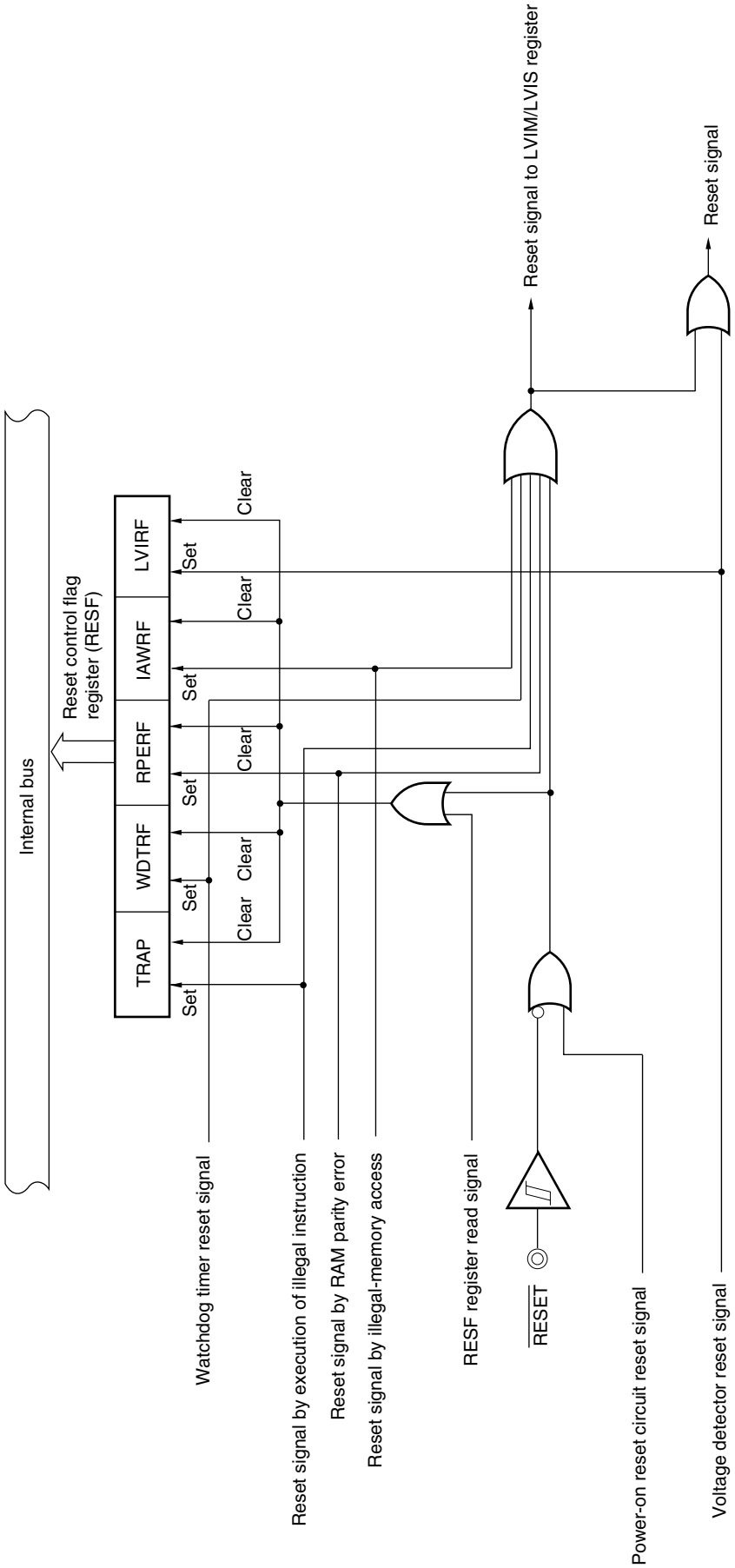
External and internal resets start program execution from the address stored at 0000H and 0001H when the reset signal is generated.

A reset is effected when a low level is input to the $\overline{\text{RESET}}$ pin, the watchdog timer overflows, or by POR and LVD circuit voltage detection, execution of illegal instruction[Note], RAM parity error or illegal-memory access, and each item of hardware is set to the status shown in Table 19-1.

**Note** The illegal instruction is generated when instruction code FFH is executed.
Reset by the illegal instruction execution not issued by emulation with the in-circuit emulator or on-chip debug emulator.

**Cautions 1. For an external reset, input a low level for 10 μs or more to the $\overline{\text{RESET}}$ pin.**
**To perform an external reset upon power application, input a low level to the $\overline{\text{RESET}}$ pin, turn power on, continue to input a low level to the pin for 10 μs or more within the operating voltage range shown in 29.4 or 30.4 AC Characteristics, and then input a high level to the pin.**
**2. During reset input, the X1 clock, XT1 clock, high-speed on-chip oscillator clock, and low-speed on-chip oscillator clock stop oscillating. External main system clock input and external subsystem clock input become invalid.**
<R> **3. The port pins become the following state because each SFR and 2nd SFR are initialized after reset.**
- **P40: High-impedance during the external reset period or reset period by the POR. High level during other types of reset or after receiving a reset signal (connected to the internal pull-up resistor).**
- **P130: Low level during the reset period or after receiving a reset signal.**
- **Ports other than P40 and P130: High-impedance during the reset period or after receiving a reset signal.**

**Figure 19-1. Block Diagram of Reset Function**




**Caution An LVD circuit internal reset does not reset the LVD circuit.**

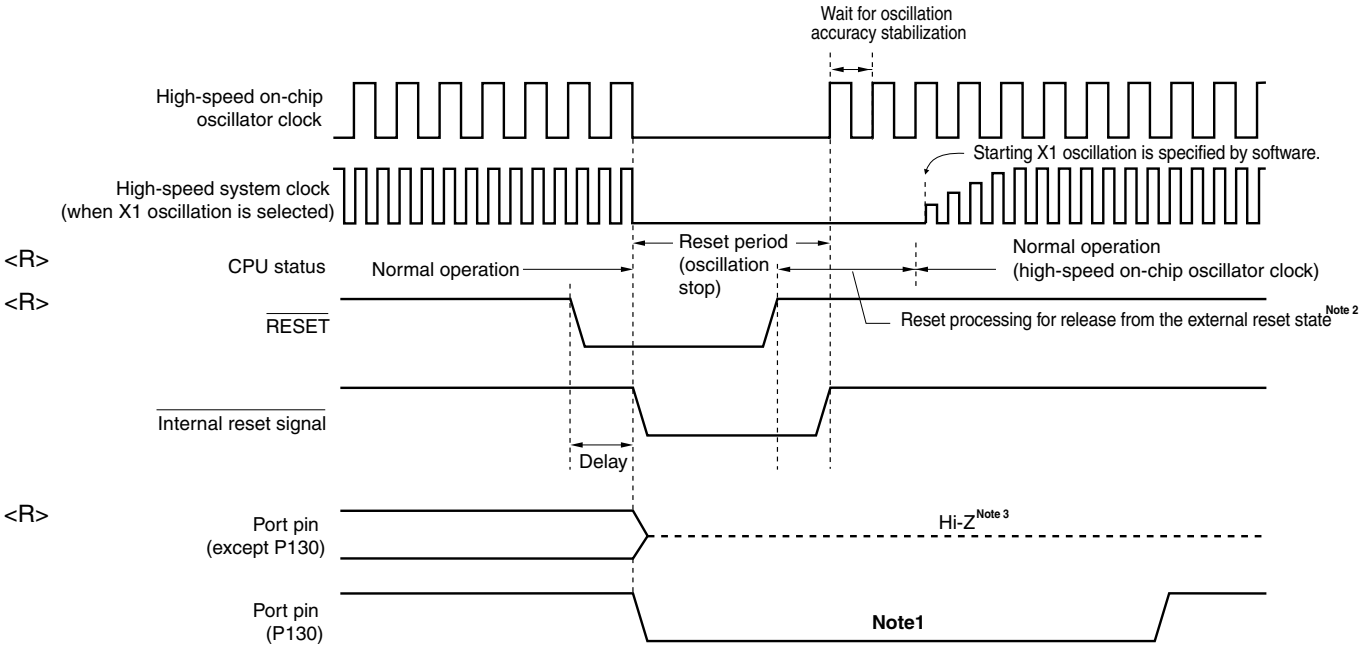
**Remarks 1.** LVIM: Voltage detection register

**2.** LVIS: Voltage detection level register

## 19.1 Timing of Reset Operation

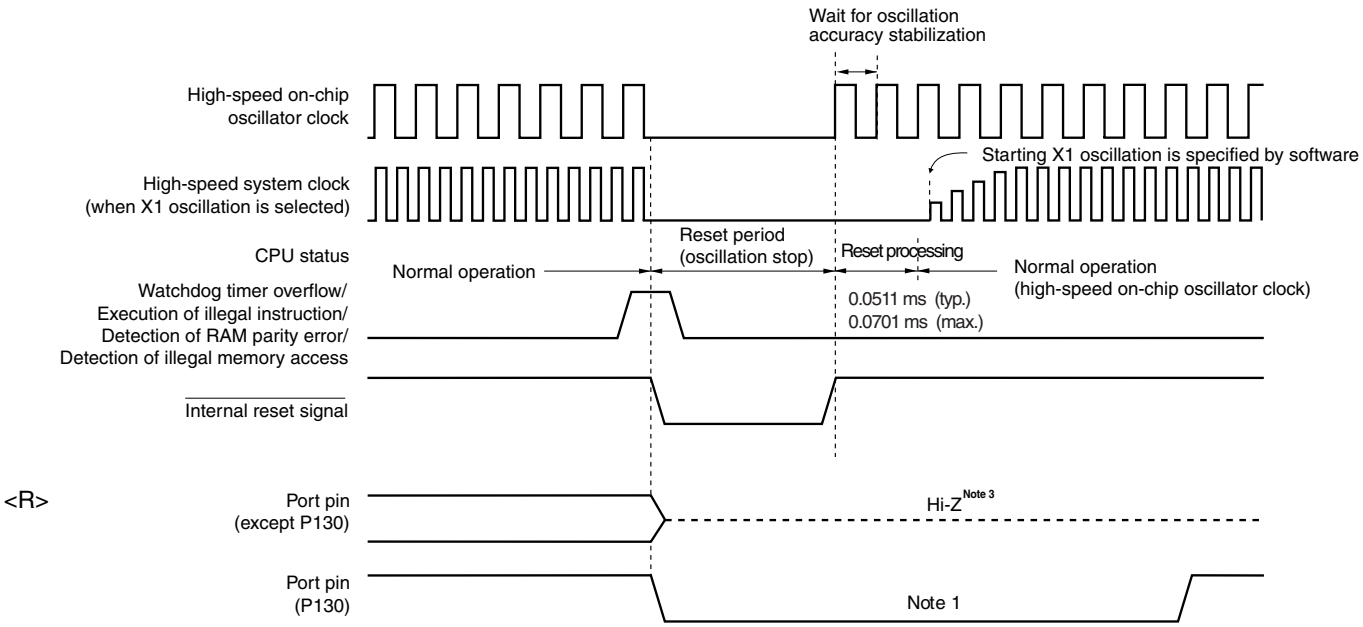
This LSI is reset by input of the low level on the $\overline{\text{RESET}}$ pin and released from the reset state by input of the high level on the $\overline{\text{RESET}}$ pin. After reset processing, execution of the program with the high-speed on-chip oscillator clock as the operating clock starts.

**Figure 19-2. Timing of Reset by $\overline{\text{RESET}}$ Input**




Release from the reset state is automatic in the case of a reset due to a watchdog timer overflow, execution of an illegal instruction, a RAM parity error, or an illegal-memory access overflow. After reset processing, execution of the program with the high-speed on-chip oscillator clock as the operating clock starts.

**Figure 19-3. Timing of Reset Due to Execution of Illegal Instruction, Watchdog Timer Overflow, RAM Parity Error, or Illegal Memory Access Overflow**




(**Notes** and **Caution** are listed on the next page.)

**Notes 1.** When P130 is set to high-level output before reset is effected, the output signal of P130 can be dummy-output as a reset signal to an external device, because P130 outputs a low level when reset is effected. To release a reset signal to an external device, set P130 to high-level output by software.

**2.** Reset times (times for release from the external reset state)

After the first release of the POR: 0.672 ms (typ.), 0.832 ms (max.) when the LVD is in use.
0.399 ms (typ.), 0.519 ms (max.) when the LVD is off.

After the second release of the POR: 0.531 ms (typ.), 0.675 ms (max.) when the LVD is in use.
0.259 ms (typ.), 0.362 ms (max.) when the LVD is off.

After power is supplied, a voltage stabilization waiting time of about 0.99 ms (typ.) and up to 2.30 ms (max.) is required before reset processing starts after release of the external reset.

<R> **3.** The state of P40 is as follows.

- High-impedance during the external reset period or reset period by the POR.
- High level during other types of reset or after receiving a reset signal (connected to the internal pull-up resistor).

<R> **Caution The watchdog timer is also reset without exception.**

Reset by POR and LVD circuit supply voltage detection is automatically released when $V_{DD} \geq V_{POR}$ or $V_{DD} \geq V_{LVD}$ after the reset. After reset processing, execution of the program with the high-speed on-chip oscillator clock as the operating clock starts. For details, see **CHAPTER 20 POWER-ON-RESET CIRCUIT** or **CHAPTER 21 VOLTAGE DETECTOR**.

**Remark** $V_{POR}$: POR power supply rise detection voltage
$V_{LVD}$: LVD detection voltage

## 19.2 States of Operation During Reset Periods

Table 19-1 shows the states of operation during reset periods. Table 19-2 shows the states of the hardware after receiving a reset signal.

**Table 19-1. States of Operation During Reset Period**

| Item | | | During Reset Period |
|---|---|---|---|
| System clock | | | Clock supply to the CPU is stopped. |
| | Main system clock | $f_{IH}$ | Operation stopped |
| | | $f_X$ | Operation stopped (the X1 and X2 pins are input port mode) |
| | | $f_{EX}$ | Clock input invalid (the pin is input port mode) |
| | Subsystem clock | $f_{XT}$ | Operation stopped (the XT1 and XT2 pins are input port mode) |
| | | $f_{EXS}$ | Clock input invalid (the pin is input port mode) |
| | $f_{IL}$ | | Operation stopped |
| CPU | | | Operation stopped |
| Code flash memory | | | Operation stopped |
| Data flash memory | | | Operation stopped |
| RAM | | | Operation stopped |
| <R> Port (latch) | | | High impedance[Note] |
| Timer array unit | | | Operation stopped |
| Real-time clock (RTC) | | | |
| 12-bit interval timer | | | |
| Watchdog timer | | | |
| Clock output/buzzer output | | | |
| A/D converter | | | |
| Serial array unit (SAU) | | | |
| Serial interface (IICA) | | | |
| Multiplier & divider, multiply-accumulator | | | |
| DMA controller | | | |
| Power-on-reset function | | | Detection operation possible |
| <R> Voltage detection function | | | Operation is possible in the case of an LVD reset and stopped in the case of other types of reset. |
| External interrupt | | | Operation stopped |
| Key interrupt function | | | |
| CRC operation function | High-speed CRC | | |
| | General-purpose CRC | | |
| RAM parity error detection function | | | |
| RAM guard function | | | |
| SFR guard function | | | |
| Illegal-memory access detection function | | | |

<R> **Note** P40 and P130 become the following state.

- P40: High-impedance during the external reset period or reset period by the POR. High level during other types of reset (connected to the internal pull-up resistor).
- P130: Low level during the reset period

(**Remark** is listed on the next page.)

**Remark** $f_{IH}$: High-speed on-chip oscillator clock
$f_X$: X1 oscillation clock
$f_{EX}$: External main system clock
$f_{XT}$: XT1 oscillation clock
$f_{EXS}$: External subsystem clock
$f_{IL}$: Low-speed on-chip oscillator clock

**Table 19-2. State of Hardware After Receiving a Reset Signal**

| Hardware | | After Reset Acknowledgment[Note] |
|---|---|---|
| Program counter (PC) | | The contents of the reset vector table (0000H, 0001H) are set. |
| Stack pointer (SP) | | Undefined |
| Program status word (PSW) | | 06H |
| RAM | Data memory | Undefined |
| | General-purpose registers | Undefined |

**Note** During reset signal generation or oscillation stabilization time wait, only the PC contents among the hardware statuses become undefined. All other hardware statuses remain unchanged after reset.

**Remark** For the state of the special function register (SFR) after receiving a reset signal, see **3.1.4 Special function egister (SFR) area** and **3.1.5 Extended special function register (2nd SFR: 2nd Special Function Register) area**.

## 19.3 Register for Confirming Reset Source

### 19.3.1 Reset control flag register (RESF)

Many internal reset generation sources exist in the RL78 microcontroller. The reset control flag register (RESF) is used to store which source has generated the reset request.

The RESF register can be read by an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input, reset by power-on-reset (POR) circuit, and reading the RESF register clear TRAP, WDTRF, RPERF, IAWRF, and LVIRF flags.

**Figure 19-4. Format of Reset Control Flag Register (RESF)**

<R> Address: FFFA8H After reset[Note 1]: R

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RESF | TRAP | 0 | 0 | WDTRF | 0 | RPERF | IAWRF | LVIRF |

| TRAP | Internal reset request by execution of illegal instruction[Note 2] |
|---|---|
| 0 | Internal reset request is not generated, or the RESF register is cleared. |
| 1 | Internal reset request is generated. |

| WDTRF | Internal reset request by watchdog timer (WDT) |
|---|---|
| 0 | Internal reset request is not generated, or the RESF register is cleared. |
| 1 | Internal reset request is generated. |

| RPERF | Internal reset request t by RAM parity |
|---|---|
| 0 | Internal reset request is not generated, or the RESF register is cleared. |
| 1 | Internal reset request is generated. |

| IAWRF | Internal reset request t by illegal-memory access |
|---|---|
| 0 | Internal reset request is not generated, or the RESF register is cleared. |
| 1 | Internal reset request is generated. |

| LVIRF | Internal reset request by voltage detector (LVD) |
|---|---|
| 0 | Internal reset request is not generated, or the RESF register is cleared. |
| 1 | Internal reset request is generated. |

**Notes 1.** The value after reset varies depending on the reset source. See **Table 19-3**.

**2.** The illegal instruction is generated when instruction code FFH is executed.
Reset by the illegal instruction execution not issued by emulation with the in-circuit emulator or on-chip debug emulator.

**Cautions 1. Do not read data by a 1-bit memory manipulation instruction.**

**2. When enabling RAM parity error resets (RPERDIS = 0), be sure to initialize the used RAM area at data access or the used RAM area + 10 bytes at execution of instruction from the RAM area. Reset generation enables RAM parity error resets (RPERDIS = 0). For details, see 22.3.3 RAM parity error detection function.**

The status of the RESF register when a reset request is generated is shown in Table 19-3.

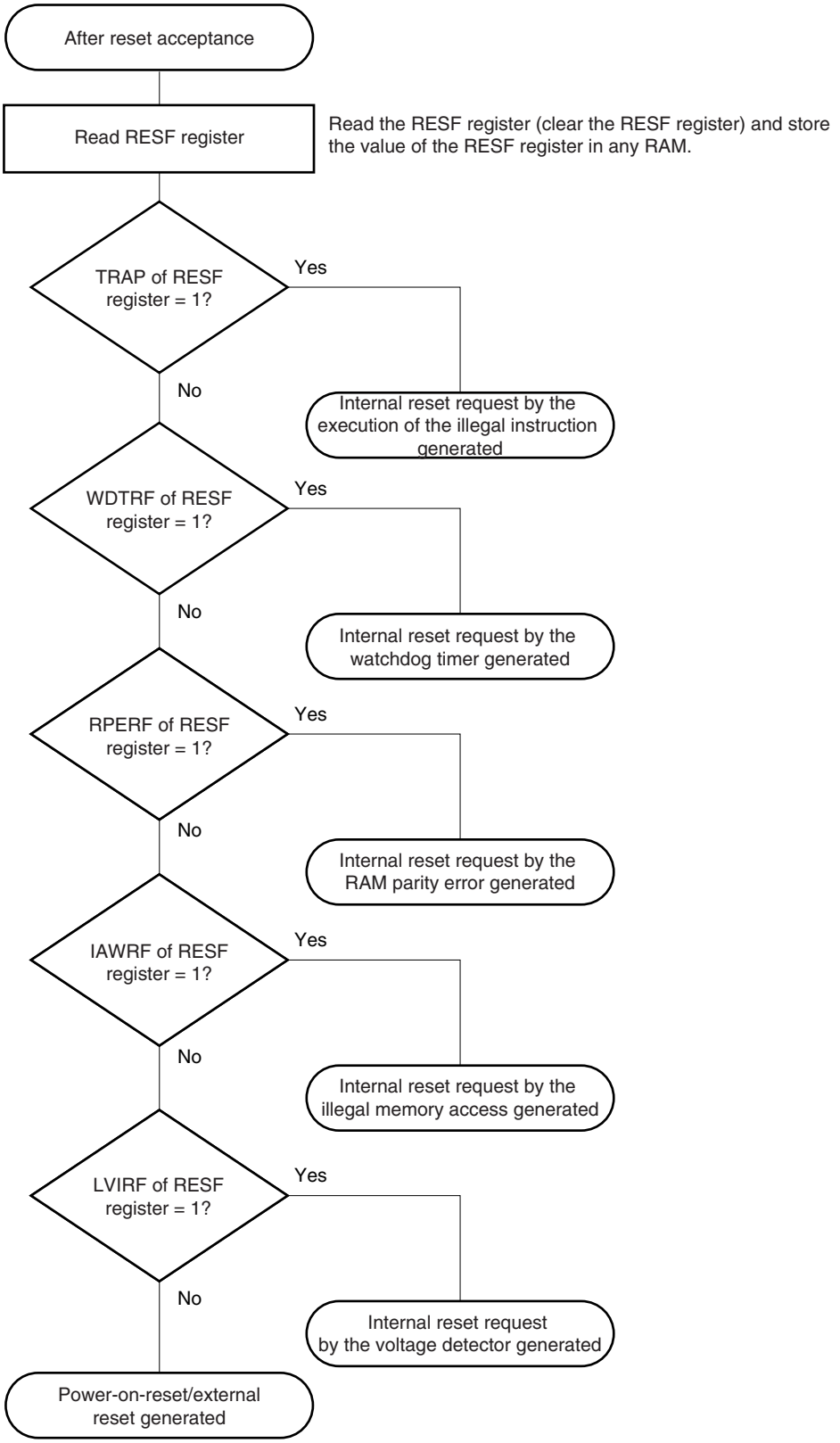
**Table 19-3. RESF Register Status When Reset Request Is Generated**

| Flag \ Reset Source | $\overline{\text{RESET}}$ Input | Reset by POR | Reset by Execution of Illegal Instruction | Reset by WDT | Reset by RAM Parity Error | Reset by Illegal-memory Access | Reset by LVD |
|---|---|---|---|---|---|---|---|
| TRAP bit | Cleared (0) | | Set (1) | Held | | | |
| WDTRF bit | | | Held | Set (1) | Held | | |
| RPERF bit | | | Held | | Set (1) | Held | |
| IAWRF bit | | | Held | | | Set (1) | Held |
| LVIRF bit | | | Held | | | | Set (1) |

<R> The RESF register is automatically cleared when it is read by an 8-bit memory manipulation instruction. Figure 19-5 shows the procedure for checking a reset source.

<R>

**Figure 19-5. Procedure for Checking Reset Source**



# CHAPTER 20 POWER-ON-RESET CIRCUIT

## 20.1 Functions of Power-on-reset Circuit

The power-on-reset circuit (POR) has the following functions.

- Generates internal reset signal at power on.

<R> The reset signal is released when the supply voltage ($V_{DD}$) exceeds the detection voltage ($V_{POR}$). Note that the reset state must be retained until the operating voltage[Note] becomes in the range defined in **29.4** or **30.4 AC Characteristics**. This is done by utilizing the voltage detection circuit or controlling the externally input reset signal.

<R> - Compares supply voltage ($V_{DD}$) and detection voltage ($V_{PDR}$), generates internal reset signal when $V_{DD} < V_{PDR}$. Note that, after the power supply is turned off, this LSI should be placed in the STOP mode, or in the reset state by utilizing the voltage detection circuit or externally input reset signal, before the operation voltage[Note] falls below the range defined in **29.4** or **30.4 AC Characteristics**. When restarting the operation, make sure that the operation voltage[Note] has returned within the range of operation.

**Note** The operating voltage range varies depending on the setting specified by the user option byte 000C2H/010C2H.

<R> **Caution If an internal reset signal is generated in the power-on-reset circuit, the reset control flag register (RESF) is cleared.**

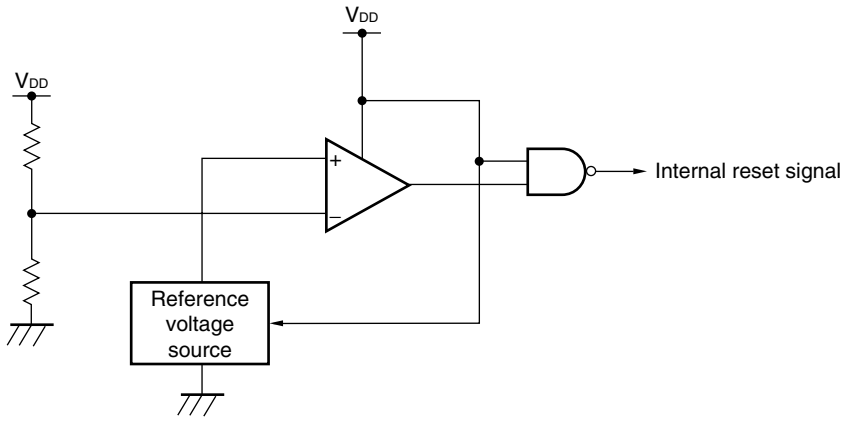
**Remarks 1.** The RL78 microcontroller incorporates multiple hardware functions that generate an internal reset signal. A flag that indicates the reset source is located in the reset control flag register (RESF) for when an internal reset signal is generated by the watchdog timer (WDT), voltage-detector (LVD), illegal instruction execution, RAM parity error, or illegal-memory access. The RESF register is not cleared to 00H and the flag is set to 1 when an internal reset signal is generated by the watchdog timer (WDT), voltage-detector (LVD), illegal instruction execution, RAM parity error, or illegal-memory access.
For details of the RESF register, see **CHAPTER 19 RESET FUNCTION**.

<R> **2.** $V_{POR}$: POR power supply rise detection voltage
$V_{PDR}$: POR power supply fall detection voltage
For details, see **29.6.3** or **30.6.3 POR circuit characteristics**.

## 20.2 Configuration of Power-on-reset Circuit

The block diagram of the power-on-reset circuit is shown in Figure 20-1.

**Figure 20-1. Block Diagram of Power-on-reset Circuit**

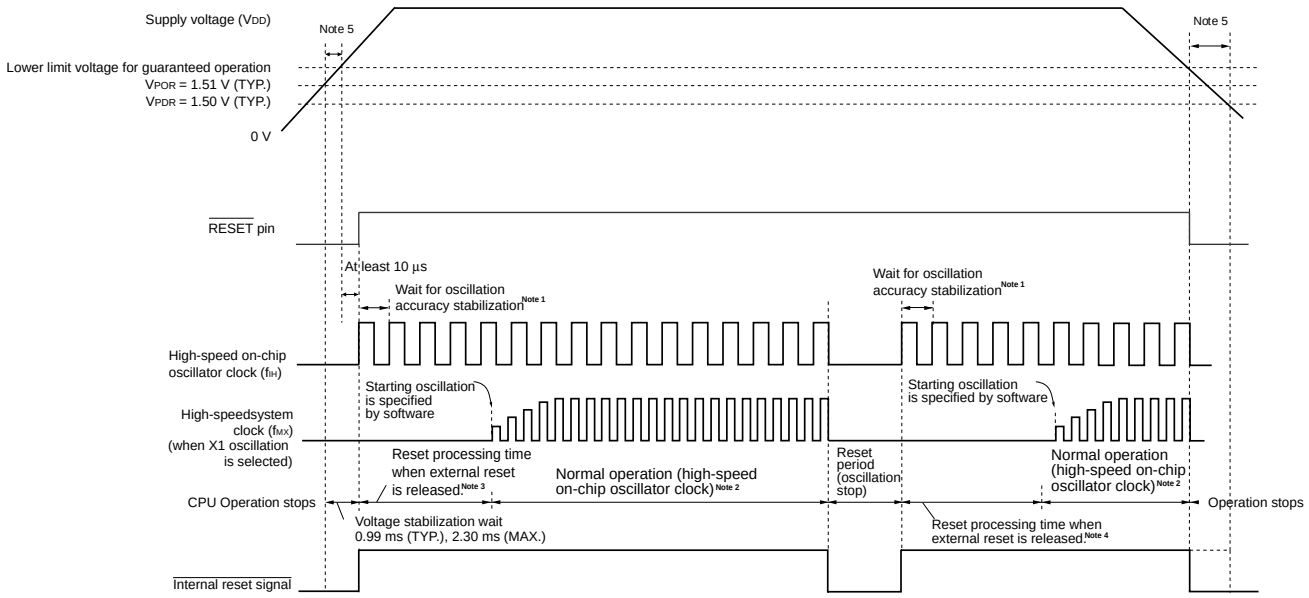



## 20.3 Operation of Power-on-reset Circuit

The timing of generation of the internal reset signal by the power-on-reset circuit and voltage detector is shown below.

**Figure 20-2. Timing of Generation of Internal Reset Signal by Power-on-reset Circuit and Voltage Detector (1/3)**

<R> (1) When the externally input reset signal on the $\overline{\text{RESET}}$ pin is used




**Notes 1.** The internal reset processing time includes the oscillation accuracy stabilization time of the high-speed on-chip oscillator clock.

**2.** The high-speed on-chip oscillator clock and a high-speed system clock or subsystem clock can be selected as the CPU clock. To use the X1 clock, use the oscillation stabilization time counter status register (OSTC) to confirm the lapse of the oscillation stabilization time. To use the XT1 clock, use the timer function for confirmation of the lapse of the stabilization time.

**3.** The time until normal operation starts includes the following reset processing time when the external reset is released (after the first release of POR) after the $\overline{\text{RESET}}$ signal is driven high (1) as well as the voltage stabilization wait time after VPOR (1.51 V, typ.) is reached.

Reset processing time when the external reset is released is shown below.

After the first release of POR: 0.672 ms (typ.), 0.832 ms (max.) (when the LVD is in use)
0.399 ms (typ.), 0.519 ms (max.) (when the LVD is off)

**4.** Reset processing time when the external reset is released after the second release of POR is shown below.

After the second release of POR: 0.531 ms (typ.), 0.675 ms (max.) (when the LVD is in use)
0.259 ms (typ.), 0.362 ms (max.) (when the LVD is off)

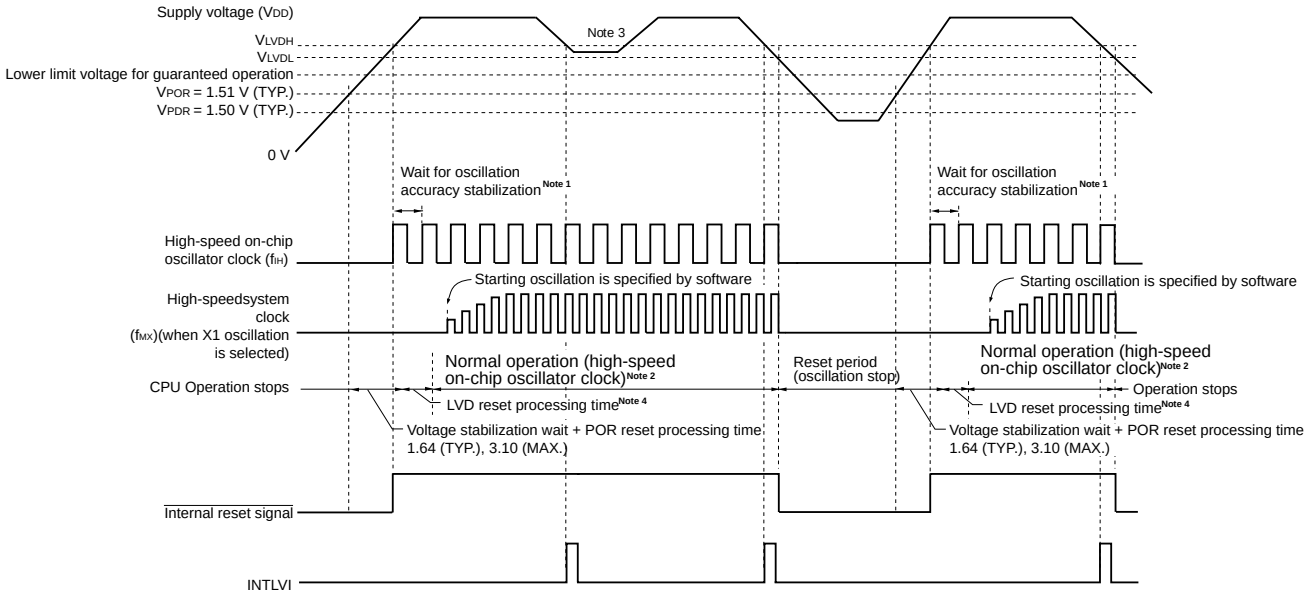
<R> **5.** After power is supplied, the reset state must be retained until the operating voltage becomes in the range defined in **29.4** or **30.4 AC Characteristics**. This is done by controlling the externally input reset signal. After power supply is turned off, this LSI should be placed in the STOP mode, or in the reset state by utilizing the voltage detection circuit or externally input reset signal, before the voltage falls below the operating range. When restarting the operation, make sure that the operation voltage has returned within the range of operation.

**Remark** VPOR: POR power supply rise detection voltage
VPDR: POR power supply fall detection voltage

<R> **Caution For power-on reset, be sure to use the externally input reset signal on the $\overline{\text{RESET}}$ pin when the LVD is off. For details, see CHAPTER 21 VOLTAGE DETECTOR.**

**Figure 20-2. Timing of Generation of Internal Reset Signal by Power-on-reset Circuit and Voltage Detector (2/3)**

<R> (2) LVD interrupt & reset mode (option byte 000C1: LVIMDS1, LVIMDS0 = 1, 0)




**Notes 1.** The internal reset processing time includes the oscillation accuracy stabilization time of the high-speed on-chip oscillator clock.

**2.** The high-speed on-chip oscillator clock and a high-speed system clock or subsystem clock can be selected as the CPU clock. To use the X1 clock, use the oscillation stabilization time counter status register (OSTC) to confirm the lapse of the oscillation stabilization time. To use the XT1 clock, use the timer function for confirmation of the lapse of the stabilization time.

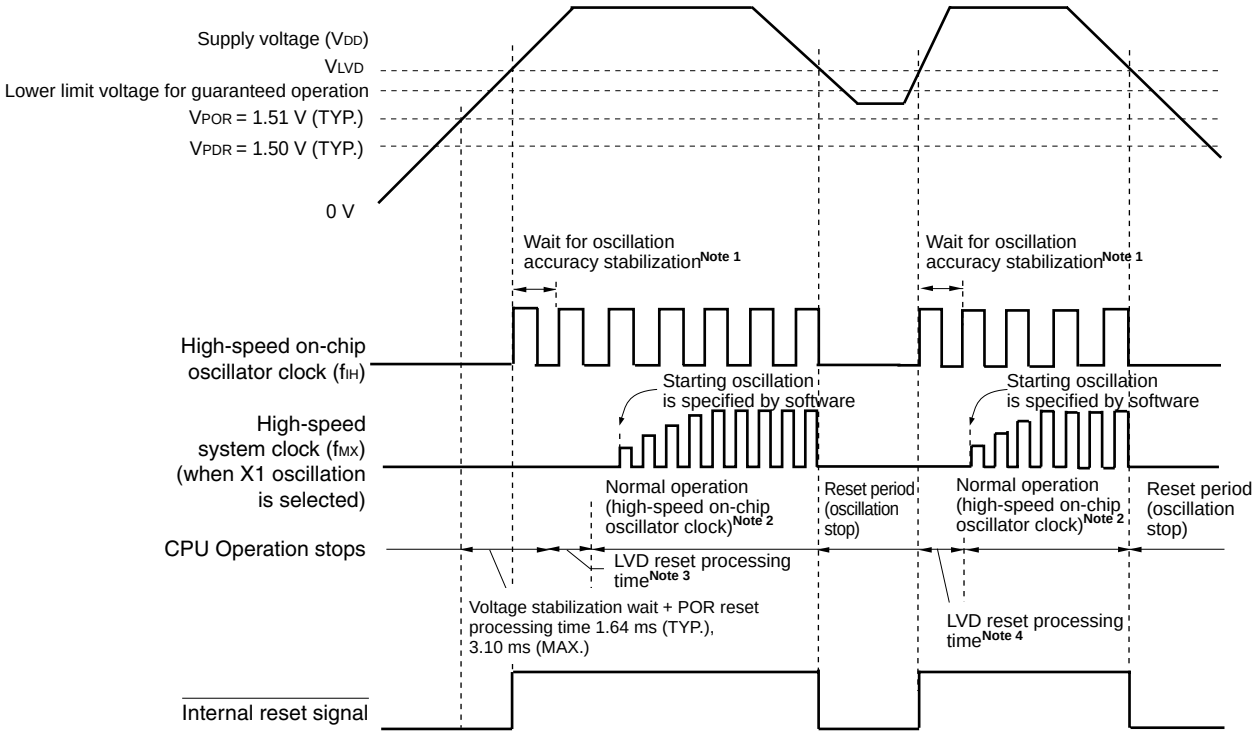
**3.** After the interrupt request signal (INTLVI) is generated, the LVIL and LVIMD bits of the voltage detection level register (LVIS) are automatically set to 1. After INTLVI is generated, appropriate settings should be made according to **Figure 21-7 Processing Procedure After an Interrupt Is Generated** and **Figure 21-8 Initial Setting of Interrupt and Reset Mode**, taking into consideration that the supply voltage might return to the high voltage detection level ($V_{LVDH}$) or higher without falling below the low voltage detection level ($V_{LVDL}$).

**4.** The time until normal operation starts includes the following LVD reset processing time after the LVD detection level ($V_{LVDH}$) is reached as well as the voltage stabilization wait + POR reset processing time after the $V_{POR}$ (1.51 V, typ.) is reached.
LVD reset processing time: 0 ms to 0.0701 ms (max.)

**Remark** $V_{LVDH}$, $V_{LVDL}$: LVD detection voltage
$V_{POR}$: POR power supply rise detection voltage
$V_{PDR}$: POR power supply fall detection voltage

**Figure 20-2. Timing of Generation of Internal Reset Signal by Power-on-reset Circuit and Voltage Detector (3/3)**

<R> (3) LVD reset mode (option byte 000C1H: LVIMDS1 = 1, LVIMDS0 = 1)




**Notes 1.** The internal reset processing time includes the oscillation accuracy stabilization time of the high-speed on-chip oscillator clock.

**2.** The high-speed on-chip oscillator clock and a high-speed system clock or subsystem clock can be selected as the CPU clock. To use the X1 clock, use the oscillation stabilization time counter status register (OSTC) to confirm the lapse of the oscillation stabilization time. To use the XT1 clock, use the timer function for confirmation of the lapse of the stabilization time.

**3.** The time until normal operation starts includes the following LVD reset processing time after the LVD detection level ($V_{LVD}$) is reached as well as the voltage stabilization wait + POR reset processing time after the $V_{POR}$ (1.51 V, typ.) is reached.
LVD reset processing time: 0 ms to 0.0701 ms (max.)

**4.** When the power supply voltage is below the lower limit for operation and the power supply voltage is then restored after an internal reset is generated only by the voltage detector (LVD), the following LVD reset processing time is required after the LVD detection level ($V_{LVD}$) is reached.
LVD reset processing time: 0.0511 ms (typ.), 0.0701 ms (max.)

**Remarks 1.** $V_{LVDH}$, $V_{LVDL}$: LVD detection voltage
$V_{POR}$: POR power supply rise detection voltage
$V_{PDR}$: POR power supply fall detection voltage

<R> **2.** When the LVD interrupt mode is selected (option byte 000C1H: LVIMD1 = 0, LVIMD0 = 1), the time until normal operation starts after power is turned on is the same as the time specified in Note 3 of Figure 20-2 (3).

# CHAPTER 21 VOLTAGE DETECTOR

## 21.1 Functions of Voltage Detector

The operation mode and detection voltages ($V_{LVDH}$, $V_{LVDL}$, $V_{LVD}$) for the voltage detector is set by using the option byte (000C1H).

The voltage detector (LVD) has the following functions.

<R> • The LVD circuit compares the supply voltage ($V_{DD}$) with the detection voltage ($V_{LVDH}$, $V_{LVDL}$, $V_{LVD}$), and generates an internal reset or interrupt request signal.

<R> • The detection level for the power supply detection voltage ($V_{LVDH}$, $V_{LVDL}$, $V_{LVD}$) can be selected by using the option byte as one of 12 levels (for details, see **CHAPTER 24 OPTION BYTE**).

• Operable in STOP mode.

• After power is supplied, the reset state must be retained until the operating voltage becomes in the range defined in **29.4** or **30.4 AC Characteristics**. This is done by utilizing the voltage detector or controlling the externally input reset signal. After the power supply is turned off, this LSI should be placed in the STOP mode, or placed in the reset state by utilizing the voltage detector or controlling the externally input reset signal before the voltage falls below the operating range. The range of operating voltage varies with the setting of the user option byte (000C2H or 010C2H).

**(a) Interrupt & reset mode (option byte LVIMDS1, LVIMDS0 = 1, 0)**

<R> The two detection voltages ($V_{LVDH}$, $V_{LVDL}$) are selected by the option byte 000C1H. The high-voltage detection level ($V_{LVDH}$) is used for releasing resets and generating interrupts. The low-voltage detection level ($V_{LVDL}$) is used for generating resets.

**(b) Reset mode (option byte LVIMDS1, LVIMDS0 = 1, 1)**

The detection voltage ($V_{LVD}$) selected by the option byte 000C1H is used for generating/releasing resets.

**(c) Interrupt mode (option byte LVIMDS1, LVIMDS0 = 0, 1)**

The detection voltage ($V_{LVD}$) selected by the option byte 000C1H is used for releasing resets/generating interrupts.

The reset and internal interrupt signals are generated in each mode as follows.

<R>

| Interrupt & Reset Mode (LVIMDS1, LVIMDS0 = 1, 0) | Reset Mode (LVIMDS1, LVIMDS0 = 1, 1) | Interrupt Mode (LVIMDS1, LVIMDS0 = 0, 1) |
|---|---|---|
| Generates an interrupt request signal by detecting $V_{DD} < V_{LVDH}$ when the operating voltage falls, and an internal reset by detecting $V_{DD} < V_{LVDL}$.<br>Releases an internal reset by detecting $V_{DD} \geq V_{LVDH}$. | Releases an internal reset by detecting $V_{DD} \geq V_{LVD}$.<br>Generates an interrupt request signal by detecting $V_{DD} < V_{LVD}$. | Releases an internal reset by detecting $V_{DD} \geq V_{LVD}$ at power on after the first release of the POR.<br>Generates an interrupt request signal by detecting $V_{DD} < V_{LVD}$ or $V_{DD} \geq V_{LVD}$ at power on after the second release of the POR. |

While the voltage detector is operating, whether the supply voltage is more than or less than the detection level can be checked by reading the voltage detection flag (LVIF: bit 0 of the voltage detection register (LVIM)).

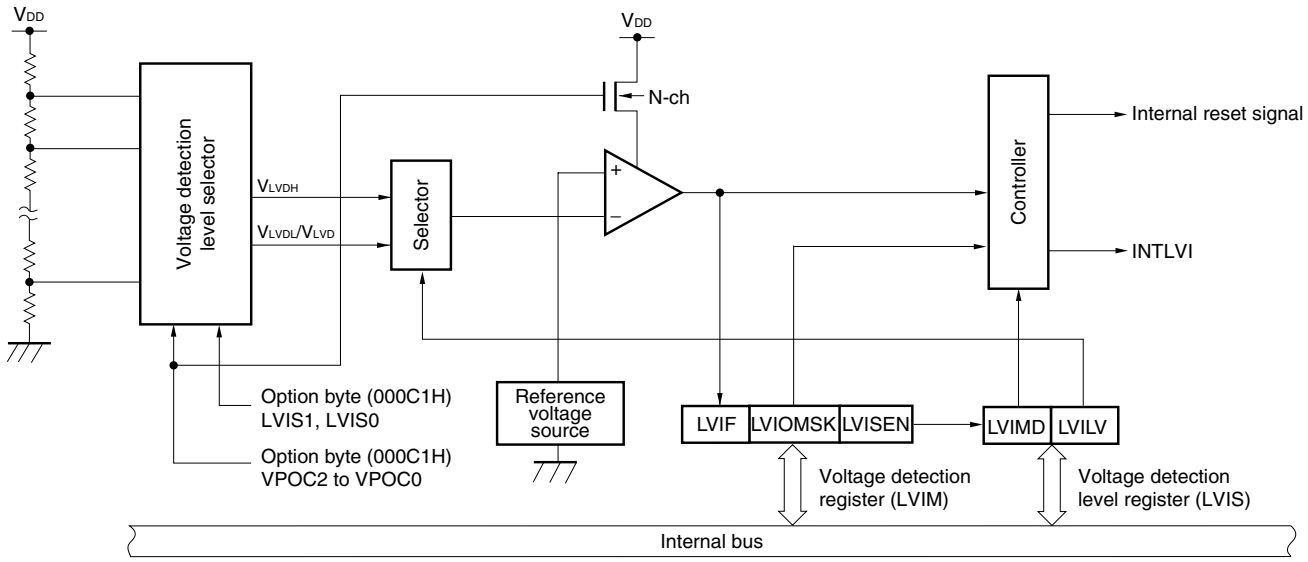
Bit 0 (LVIRF) of the reset control flag register (RESF) is set to 1 if reset occurs. For details of the RESF register, see **CHAPTER 19 RESET FUNCTION**.

## 21.2 Configuration of Voltage Detector

The block diagram of the voltage detector is shown in Figure 21-1.

<R>

**Figure 21-1. Block Diagram of Voltage Detector**




## 21.3 Registers Controlling Voltage Detector

The voltage detector is controlled by the following registers.

- Voltage detection register (LVIM)
- Voltage detection level register (LVIS)

### 21.3.1 Voltage detection register (LVIM)

This register is used to specify whether to enable or disable rewriting the voltage detection level register (LVIS), as well as to check the LVD output mask status.

This register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 21-2. Format of Voltage Detection Register (LVIM)**

Address: FFFA9H After reset: 00H[Note 1] R/W[Note 2]

| Symbol | <7> | 6 | 5 | 4 | 3 | 2 | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| LVIM | LVISEN[Note 3] | 0 | 0 | 0 | 0 | 0 | LVIOMSK | LVIF |

| LVISEN[Note 3] | Specification of whether to enable or disable rewriting the voltage detection level register (LVIS) |
|---|---|
| <R> 0 | Disabling of rewriting the LVIS register (LVIOMSK = 0 (Mask of LVD output is invalid) |
| <R> 1 | Enabling of rewriting the LVIS register (LVIOMSK = 1 (Mask of LVD output is valid) |

| LVIOMSK | Mask status flag of LVD output |
|---|---|
| <R> 0 | Mask of LVD output is invalid |
| <R> 1 | Mask of LVD output is valid[Note 4] |

| LVIF | Voltage detection flag |
|---|---|
| <R> 0 | Supply voltage ($V_{DD}$) ≥ detection voltage ($V_{LVD}$), or when LVD is off |
| 1 | Supply voltage ($V_{DD}$) < detection voltage ($V_{LVD}$) |

**Notes 1.** The reset value changes depending on the reset source.
If the LVIS register is reset by LVD, it is not reset but holds the current value. In other reset, LVISEN is cleared to 0.

**2.** Bits 0 and 1 are read-only.

<R> **3.** LVISEN can only be set in the interrupt & reset mode (option byte LVIMDS1, LVIMDS0 = 1, 0). Do not change the initial value in other modes.

**4.** LVIOMSK bit is only automatically set to “1” when the interrupt & reset mode is selected (option byte LVIMDS1, LVIMDS0 = 1, 0) and reset or interrupt by LVD is masked.
- Period during LVISEN = 1
- Waiting period from the time when LVD interrupt is generated until LVD detection voltage becomes stable
- Waiting period from the time when the value of LVILV bit changes until LVD detection voltage becomes stable

### 21.3.2 Voltage detection level register (LVIS)

This register selects the voltage detection level.

This register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation input sets this register to 00H/01H/81H[Note 1].

**Figure 21-3. Format of Voltage Detection Level Select Register (LVIS)**

Address: FFFAAH After reset: 00H/01H/81H[Note 1] R/W

| Symbol | <7> | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| LVIS | LVIMD | 0 | 0 | 0 | 0 | 0 | 0 | LVILV |

| LVIMD[Note 2] | Operation mode of voltage detection |
|---|---|
| 0 | Interrupt mode |
| 1 | Reset mode |

| LVILV[Note 2] | LVD detection level |
|---|---|
| 0 | High-voltage detection level ($V_{LVDH}$) |
| 1 | Low-voltage detection level ($V_{LVDL}$ or $V_{LVDL}$) |

**Notes 1.** The reset value changes depending on the reset source and the setting of the option byte.
This register is not cleared (00H) by LVD reset.
The generation of reset signal other than an LVD reset sets as follows.
- When option byte LVIMDS1, LVIMDS0 = 1, 0: 00H
- When option byte LVIMDS1, LVIMDS0 = 1, 1: 81H
- When option byte LVIMDS1, LVIMDS0 = 0, 1: 01H

<R> **2.** Writing “0” can only be allowed in the interrupt & reset mode (option byte LVIMDS1, LVIMDS0 = 1, 0). Do not set LVIMD and LVILV in other cases. The value is switched automatically when reset or interrupt is generated in the interrupt & reset mode.

<R> **Cautions 1. Rewrite the value of the LVIS register according to Figures 21-7 and 21-8.**

<R> **2. Specify the LVD operation mode and detection voltage ($V_{LVDH}$, $V_{LVDL}$, $V_{LVD}$) of each mode by using the option byte 000C1H. Table 21-1 shows the format of the user option byte (000C1H/010C1H). For details about the option byte, see CHAPTER 24 OPTION BYTE.**

**Table 21-1. Format of User Option Byte (000C1H/010C1H) (1/2)**

Address: 000C1H/010C1H[Note]

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| VPOC2 | VPOC1 | VPOC0 | 1 | LVIS1 | LVIS0 | LVIMDS1 | LVIMDS0 |

<R> • LVD setting (interrupt & reset mode)

| Detection voltage | | | Option byte setting value | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $V_{LVDH}$ | | $V_{LVDL}$ | VPOC2 | VPOC1 | VPOC0 | LVIS1 | LVIS0 | Mode setting | |
| Rising edge | Falling edge | Falling edge | | | | | | LVIMDS1 | LVIMDS0 |
| 1.77 V | 1.73 V | 1.63 V | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
| 1.88 V | 1.84 V | | | | | 0 | 1 | | |
| 2.92 V | 2.86 V | | | | | 0 | 0 | | |
| 1.98 V | 1.94 V | 1.84 V | | 0 | 1 | 1 | 0 | | |
| 2.09 V | 2.04 V | | | | | 0 | 1 | | |
| 3.13 V | 3.06 V | | | | | 0 | 0 | | |
| 2.61 V | 2.55 V | 2.45 V | | 1 | 0 | 1 | 0 | | |
| 2.71 V | 2.65 V | | | | | 0 | 1 | | |
| 2.92 V | 2.86 V | 2.75 V | | 1 | 1 | 1 | 0 | | |
| 3.02 V | 2.96 V | | | | | 0 | 1 | | |
| – | | | Value other than above is setting prohibited. | | | | | | |

<R> • LVD setting (reset mode)

| Detection voltage | | Option byte setting value | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $V_{LVD}$ | | VPOC2 | VPOC1 | VPOC0 | LVIS1 | LVIS0 | Mode setting | |
| Rising edge | Falling edge | | | | | | LVIMDS1 | LVIMDS0 |
| 1.67 V | 1.63 V | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1.77 V | 1.73 V | | 0 | 0 | 1 | 0 | | |
| 1.88 V | 1.84 V | | 0 | 1 | 1 | 1 | | |
| 1.98 V | 1.94 V | | 0 | 1 | 1 | 0 | | |
| 2.09 V | 2.04 V | | 0 | 1 | 0 | 1 | | |
| 2.50 V | 2.45 V | | 1 | 0 | 1 | 1 | | |
| 2.61 V | 2.55 V | | 1 | 0 | 1 | 0 | | |
| 2.71 V | 2.65 V | | 1 | 0 | 0 | 1 | | |
| 2.81 V | 2.75 V | | 1 | 1 | 1 | 1 | | |
| 2.92 V | 2.86 V | | 1 | 1 | 1 | 0 | | |
| 3.02 V | 2.96 V | | 1 | 1 | 0 | 1 | | |
| 3.13 V | 3.06 V | | 0 | 1 | 0 | 0 | | |
| – | | Value other than above is setting prohibited. | | | | | | |

**Note** Set the same value as 000C1H to 010C1H when the boot swap operation is used because 000C1H is replaced by 010C1H.

**Remarks 1.** For details on the LVD circuit, see **CHAPTER 21 VOLTAGE DETECTOR**.

**2.** The detection voltage is a TYP. value. For details, see **29.6.4** or **30.6.4 LVD circuit characteristics**.

(**Cautions** are listed on the next page.)

**Table 21-1. Format of User Option Byte (000C1H/010C1H) (2/2)**

Address: 000C1H/010C1H[Note]

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| VPOC2 | VPOC1 | VPOC0 | 1 | LVIS1 | LVIS0 | LVIMDS1 | LVIMDS0 |

<R> • LVD setting (interrupt mode)

| Detection voltage | | Option byte setting value | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $V_{LVD}$ | | VPOC2 | VPOC1 | VPOC0 | LVIS1 | LVIS0 | Mode setting | |
| Rising edge | Falling edge | | | | | | LVIMDS1 | LVIMDS0 |
| 1.67 V | 1.63 V | 0 | 0 | 0 | 1 | 1 | 0 | 1 |
| 1.77 V | 1.73 V | | 0 | 0 | 1 | 0 | | |
| 1.88 V | 1.84 V | | 0 | 1 | 1 | 1 | | |
| 1.98 V | 1.94 V | | 0 | 1 | 1 | 0 | | |
| 2.09 V | 2.04 V | | 0 | 1 | 0 | 1 | | |
| 2.50 V | 2.45 V | | 1 | 0 | 1 | 1 | | |
| 2.61 V | 2.55 V | | 1 | 0 | 1 | 0 | | |
| 2.71 V | 2.65 V | | 1 | 0 | 0 | 1 | | |
| 2.81 V | 2.75 V | | 1 | 1 | 1 | 1 | | |
| 2.92 V | 2.86 V | | 1 | 1 | 1 | 0 | | |
| 3.02 V | 2.96 V | | 1 | 1 | 0 | 1 | | |
| 3.13 V | 3.06 V | | 0 | 1 | 0 | 0 | | |
| – | | Value other than above is setting prohibited. | | | | | | |

<R> • LVD off (use of external reset input via $\overline{\text{RESET}}$ pin)

| Detection voltage | | Option byte setting value | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $V_{LVD}$ | | VPOC2 | VPOC1 | VPOC0 | LVIS1 | LVIS0 | Mode setting | |
| Rising edge | Falling edge | | | | | | LVIMDS1 | LVIMDS0 |
| – | – | 1 | × | × | × | × | × | 1 |
| – | | Value other than above is setting prohibited. | | | | | | |

**Note** Set the same value as 000C1H to 010C1H when the boot swap operation is used because 000C1H is replaced by 010C1H.

**Cautions 1. Set bit 4 to 1.**

<R> **2. After power is supplied, the reset state must be retained until the operating voltage becomes in the range defined in 29.4 or 30.4 AC Characteristics. This is done by utilizing the voltage detector or controlling the externally input reset signal. After the power supply is turned off, this LSI should be placed in the STOP mode, or placed in the reset state by utilizing the voltage detector or controlling the externally input reset signal, before the voltage falls below the operating range.**

**The range of operating voltage varies with the setting of the user option byte (000C2H or 010C2H).**

**Remarks 1.** ×: don't care

**2.** For details on the LVD circuit, see **CHAPTER 21 VOLTAGE DETECTOR**.

**3.** The detection voltage is a TYP. value. For details, see **29.6.4** or **30.6.4 LVD circuit characteristics**.

## 21.4 Operation of Voltage Detector

### <R> 21.4.1 When used as reset mode

Specify the operation mode (the reset mode (LVIMDS1, LVIMDS0 = 1, 1)) and the detection voltage ($V_{LVD}$) by using the option byte 000C1H.

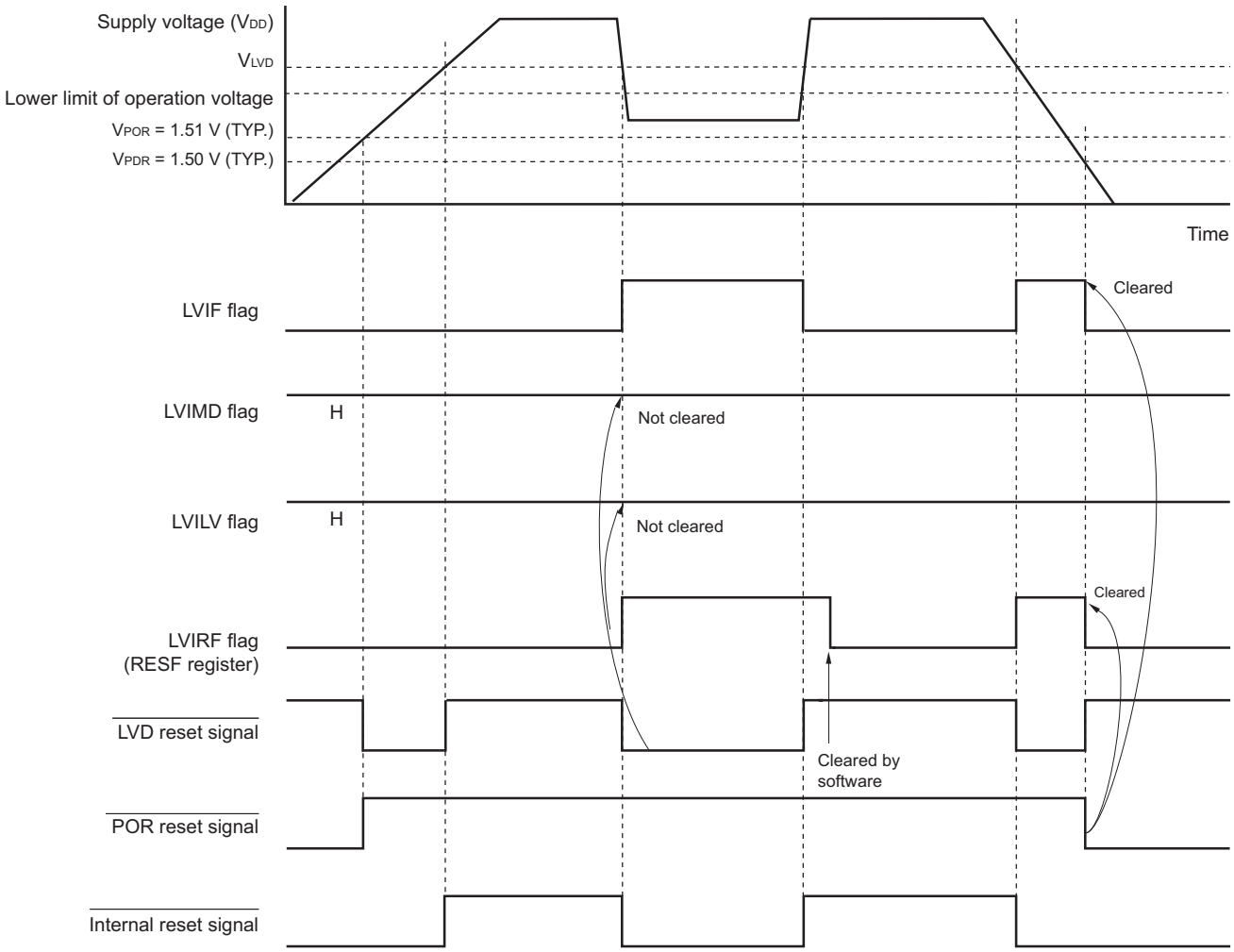
The operation is started in the following initial setting state when the reset mode is set.

- Bit 7 (LVISEN) of the voltage detection register (LVIM) is set to 0 (disable rewriting of voltage detection level register (LVIS))
- The initial value of the voltage detection level select register (LVIS) is set to 81H.
  Bit 7 (LVIMD) is 1 (reset mode).
  Bit 0 (LVILV) is 1 (voltage detection level: $V_{LVD}$).

- Operation in LVD reset mode
  In the reset mode (option byte LVIMDS1, LVIMDS0 = 1, 1), the state of an internal reset by LVD is retained until the supply voltage ($V_{DD}$) exceeds the voltage detection level ($V_{LVD}$) after power is supplied. The internal reset is released when the supply voltage ($V_{DD}$) exceeds the voltage detection level ($V_{LVD}$).
  At the fall of the operating voltage, an internal reset by LVD is generated when the supply voltage ($V_{DD}$) falls below the voltage detection level ($V_{LVD}$).

Figure 21-4 shows the timing of the internal reset signal generated in the LVD reset mode.

<R> **Figure 21-4. Timing of Voltage Detector Internal Reset Signal Generation (Option Byte LVIMDS1, LVIMDS0 = 1, 1)**




**Remark** VPOR: POR power supply rise detection voltage
VPDR: POR power supply fall detection voltage

### <R> 21.4.2 When used as interrupt mode

Specify the operation mode (the interrupt mode (LVIMDS1, LVIMDS0 = 0, 1)) and the detection voltage ($V_{LVD}$) by using the option byte 000C1H.

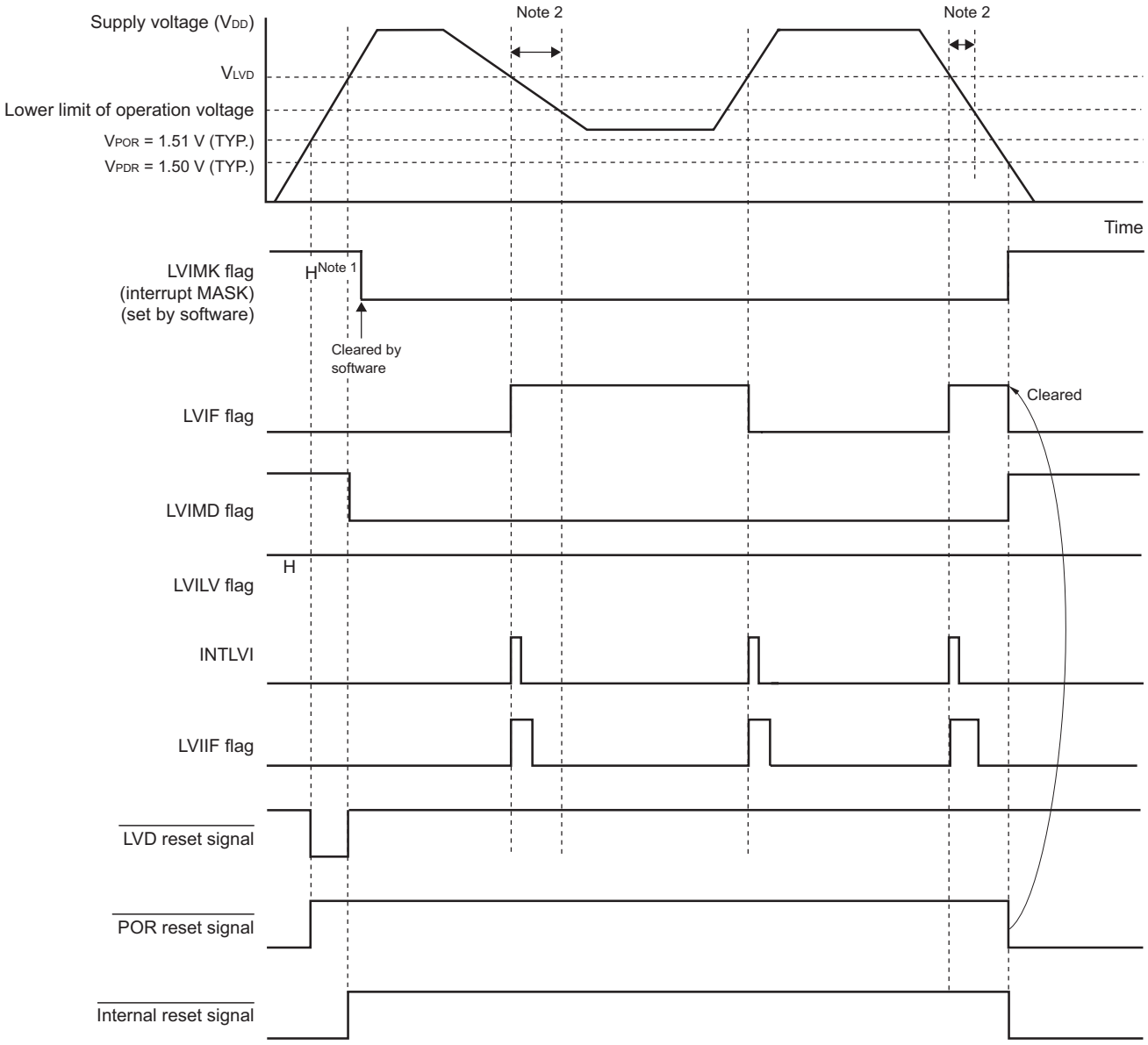
The operation is started in the following initial setting state when the interrupt mode is set.

- Bit 7 (LVISEN) of the voltage detection register (LVIM) is set to 0 (disable rewriting of voltage detection level register (LVIS))
- The initial value of the voltage detection level select register (LVIS) is set to 01H.
  Bit 7 (LVIMD) is 0 (interrupt mode).
  Bit 0 (LVILV) is 1 (low-voltage detection level: $V_{LVD}$).

- Operation in LVD interrupt mode
  In the interrupt mode (option byte LVIMDS1, LVIMDS0 = 0, 1), the state of an internal reset by LVD is retained until the supply voltage ($V_{DD}$) exceeds the voltage detection level ($V_{LVD}$) after power is supplied (after the first release of the POR). The internal reset is released when the supply voltage ($V_{DD}$) exceeds the voltage detection level ($V_{LVD}$).
  An interrupt request signal by LVD (INTLVD) is generated, when the supply voltage ($V_{DD}$) falls below the voltage detection level ($V_{LVD}$) or when the supply voltage ($V_{DD}$) exceeds the voltage detection level ($V_{LVD}$) after the second release of the POR. When the voltage falls, this LSI should be placed in the STOP mode, or placed in the reset state by controlling the externally input reset signal, before the voltage falls below the operating voltage range defined in **29.4** or **30.4 AC characteristics**. When restarting the operation, make sure that the operation voltage has returned within the range of operation.

Figure 21-5 shows the timing of the interrupt request signal generated in the LVD interrupt mode.

<R> **Figure 21-5. Timing of Voltage Detector Internal Interrupt Signal Generation (Option Byte LVIMDS1, LVIMDS0 = 0, 1)**




**Notes 1.** The LVIMK flag is set to “1” by reset signal generation.

<R> **2.** When the voltage falls, this LSI should be placed in the STOP mode, or placed in the reset state by controlling the externally input reset signal, before the voltage falls below the operating voltage range defined in **29.4** or **30.4 AC characteristics**. When restarting the operation, make sure that the operation voltage has returned within the range of operation.

**Remark** VPOR: POR power supply rise detection voltage
VPDR: POR power supply fall detection voltage

### <R> 21.4.3 When used as interrupt & reset mode

Specify the operation mode (the interrupt & reset (LVIMDS1, LVIMDS0 = 1, 0)) and the detection voltage ($V_{LVDH}$, $V_{LVDL}$) by using the option byte 000C1H.

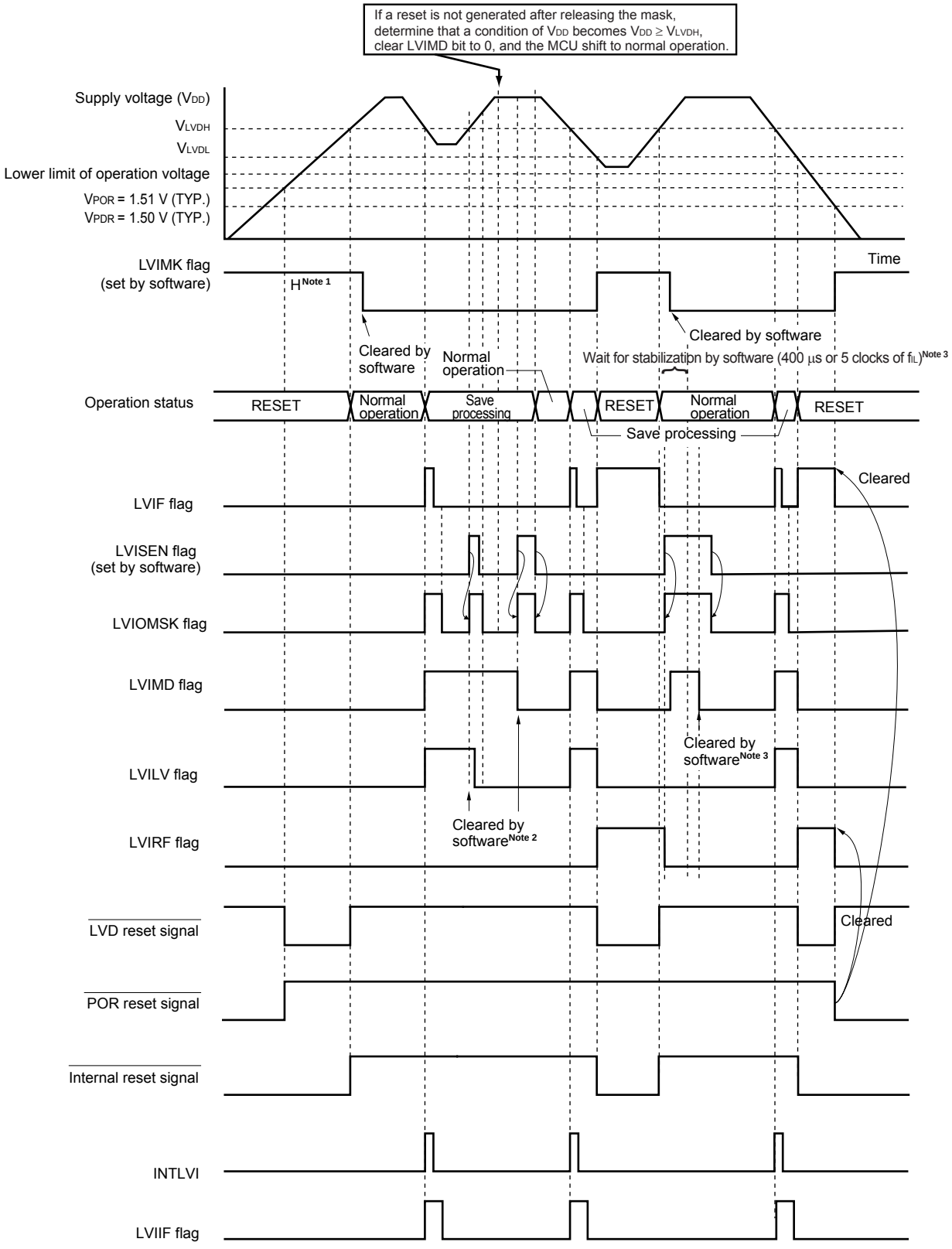
The operation is started in the following initial setting state when the interrupt & reset mode is set.

- Bit 7 (LVISEN) of the voltage detection register (LVIM) is set to 0 (disable rewriting of voltage detection level register (LVIS))
- The initial value of the voltage detection level select register (LVIS) is set to 00H.
  Bit 7 (LVIMD) is 0 (interrupt mode).
  Bit 0 (LVILV) is 0 (high-voltage detection level: $V_{LVDH}$).

- Operation in LVD interrupt & reset mode
  In the interrupt & reset mode (option byte LVIMDS1, LVIMDS0 = 1, 0), the state of an internal reset by LVD is retained until the supply voltage ($V_{DD}$) exceeds the high-voltage detection level ($V_{LVDH}$) after power is supplied. The internal reset is released when the supply voltage ($V_{DD}$) exceeds the high-voltage detection level ($V_{LVDH}$).
  At the fall of the operating voltage, an interrupt request signal (INTLVD) by LVD is generated and arbitrary save processing is performed when the supply voltage ($V_{DD}$) falls below the high-voltage detection level ($V_{LVDH}$). After that, an internal reset by LVD is generated when the supply voltage ($V_{DD}$) falls below the low-voltage detection level ($V_{LVDL}$). However, after INTLVD is generated, an interrupt request signal is not generated even if the supply voltage becomes equal to or higher than the high-voltage detection voltage ($V_{LVDH}$) without falling below the low-voltage detection voltage ($V_{LVDL}$).
  To use the LVD reset & interrupt mode, perform the processing according to **Figure 21-7 Processing Procedure After an Interrupt Is Generated** and **Figure 21-8 Initial Setting of Interrupt and Reset Mode**.

Figure 21-6 shows the timing of the internal reset signal and interrupt signal generated in the LVD interrupt & reset mode.

<R> **Figure 21-6. Timing of Voltage Detector Reset Signal and Interrupt Signal Generation (Option Byte LVIMDS1, LVIMDS0 = 1, 0) (1/2)**




(**Notes** and **Remark** are listed on the next page.)

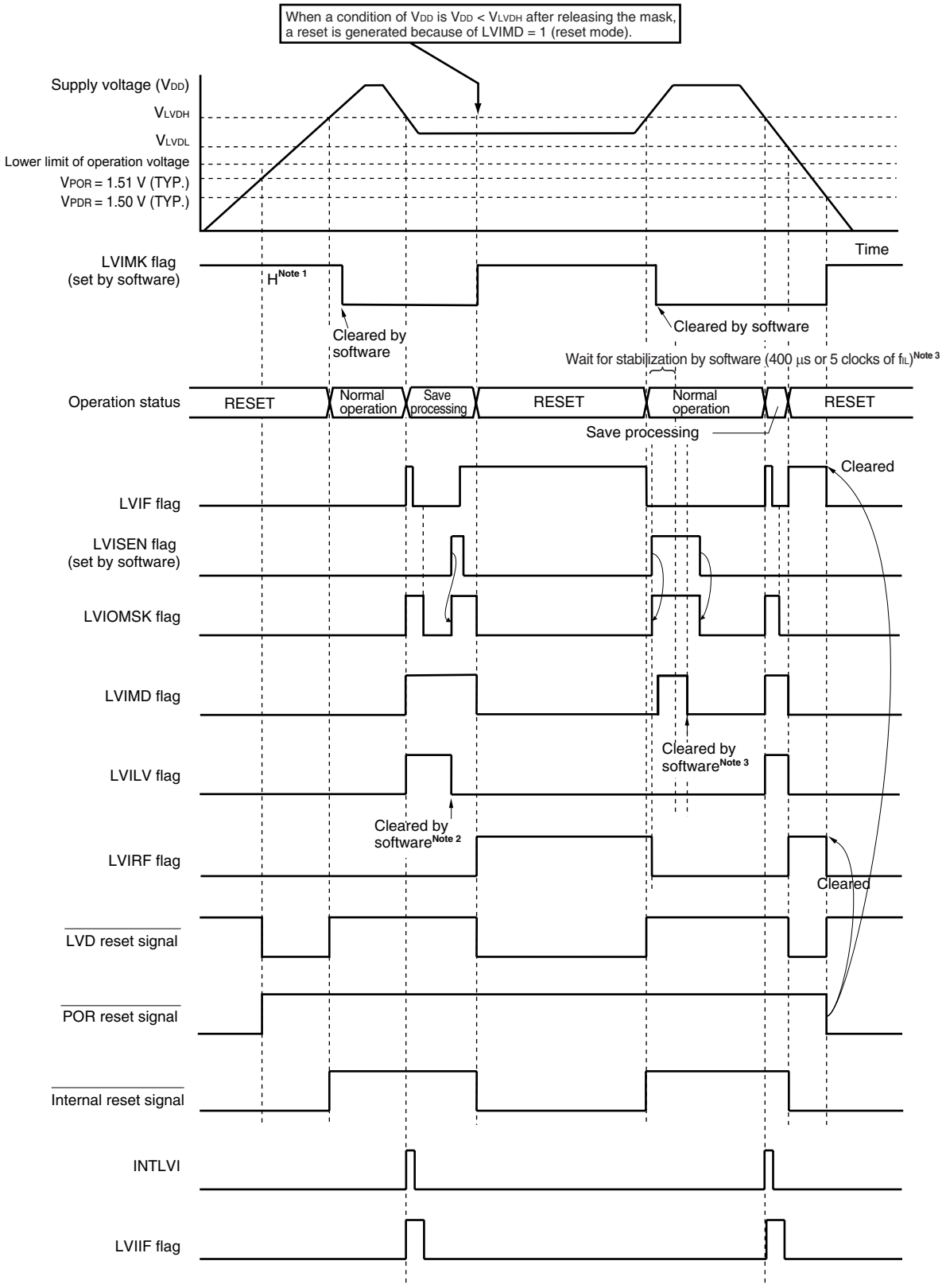
**Notes 1.** The LVIMK flag is set to “1” by reset signal generation.

**2.** After an interrupt is generated, perform the processing according to **Figure 21-7 Processing Procedure After an Interrupt Is Generated**.

**3.** After a reset is released, perform the processing according to **Figure 21-8 Initial Setting of Interrupt and Reset Mode**.

**Remark** $V_{POR}$: POR power supply rise detection voltage
$V_{PDR}$: POR power supply fall detection voltage

<R> **Figure 21-6. Timing of Voltage Detector Reset Signal and Interrupt Signal Generation (Option Byte LVIMDS1, LVIMDS0 = 1, 0) (2/2)**




(**Notes** and **Remark** are listed on the next page.)

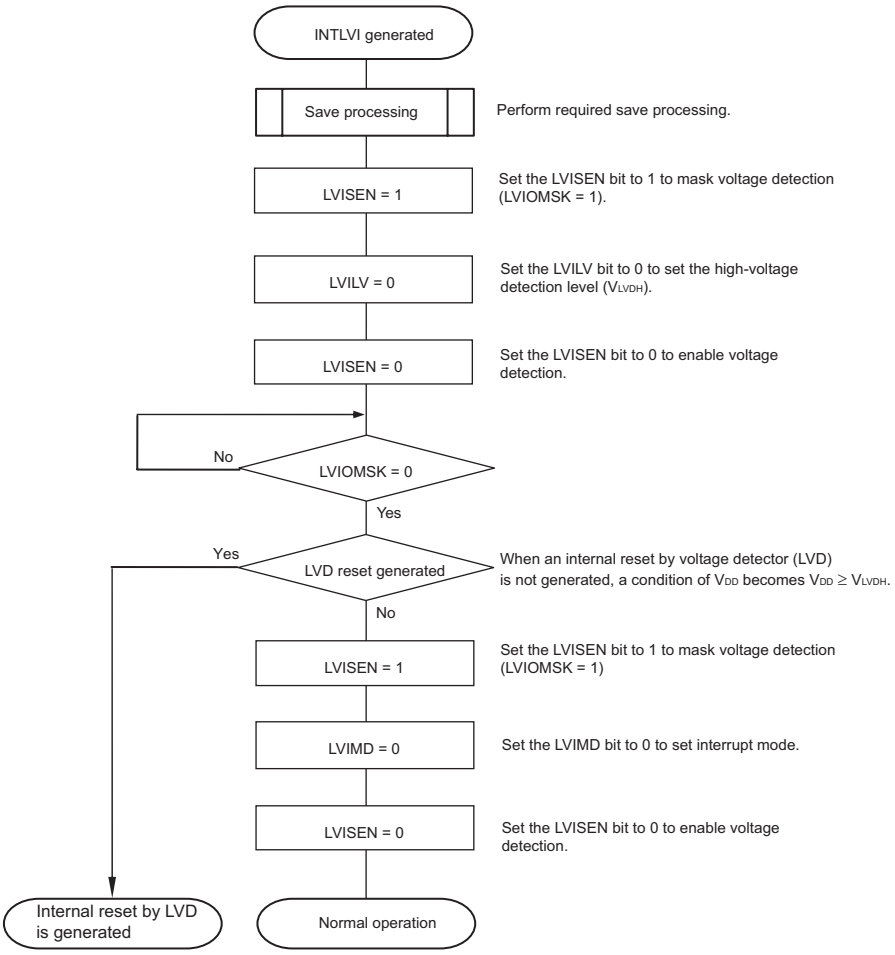
**Notes 1.** The LVIMK flag is set to “1” by reset signal generation.

**2.** After an interrupt is generated, perform the processing according to **Figure 21-7 Processing Procedure After an Interrupt Is Generated**.

**3.** After a reset is released, perform the processing according to **Figure 21-8 Initial Setting of Interrupt and Reset Mode**.

**Remark** $V_{POR}$: POR power supply rise detection voltage
$V_{PDR}$: POR power supply fall detection voltage

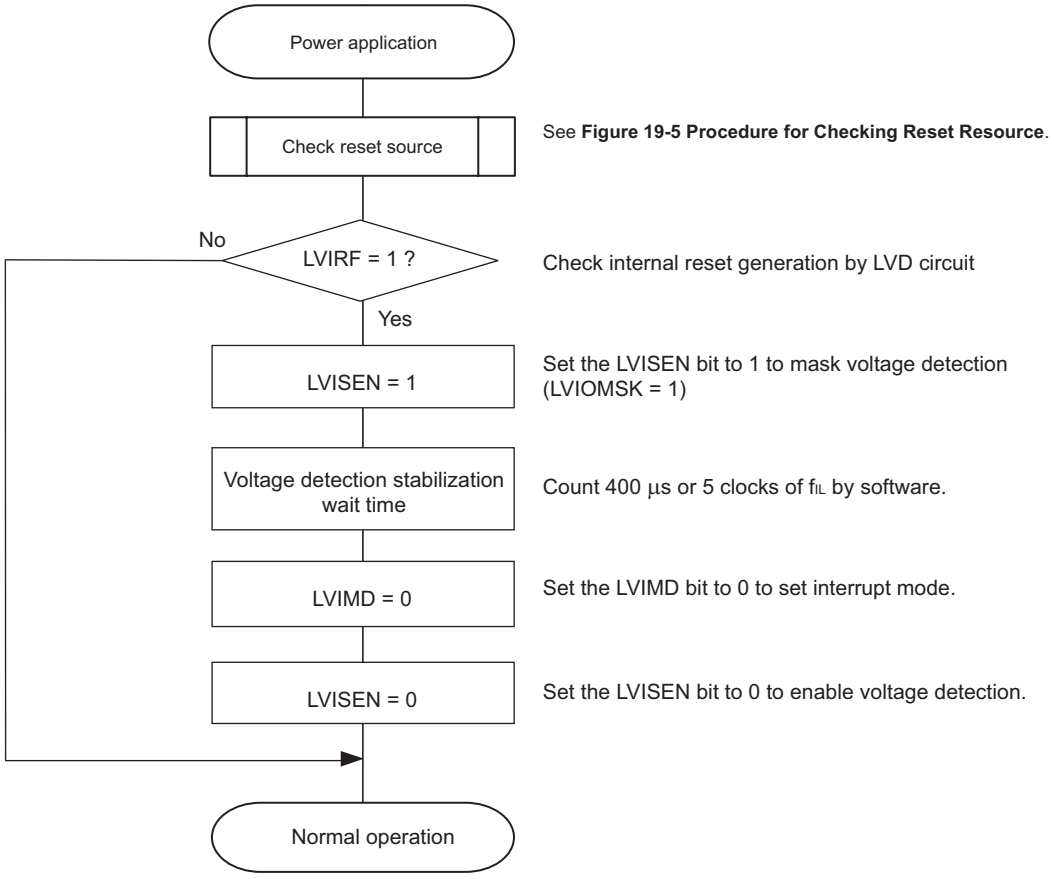
<R> **Figure 21-7. Processing Procedure After an Interrupt Is Generated**



When setting an interrupt and reset mode (LVIMDS1, LVIMDS0 = 1, 0), voltage detection stabilization wait time for 400 μs or 5 clocks of $f_{IL}$ is necessary after LVD reset is released (LVIRF = 1). After waiting until voltage detection stabilizes, (0) clear the LVIMD bit for initialization. While voltage detection stabilization wait time is being counted and when the LVIMD bit is rewritten, set LVISEN to 1 to mask a reset or interrupt generation by LVD.

Figure 21-8 shows the procedure for initial setting of interrupt and reset mode.

<R> **Figure 21-8. Initial Setting of Interrupt and Reset Mode**




**Remark** $f_{IL}$: Low-speed on-chip oscillator clock frequency

## 21.5 Cautions for Voltage Detector

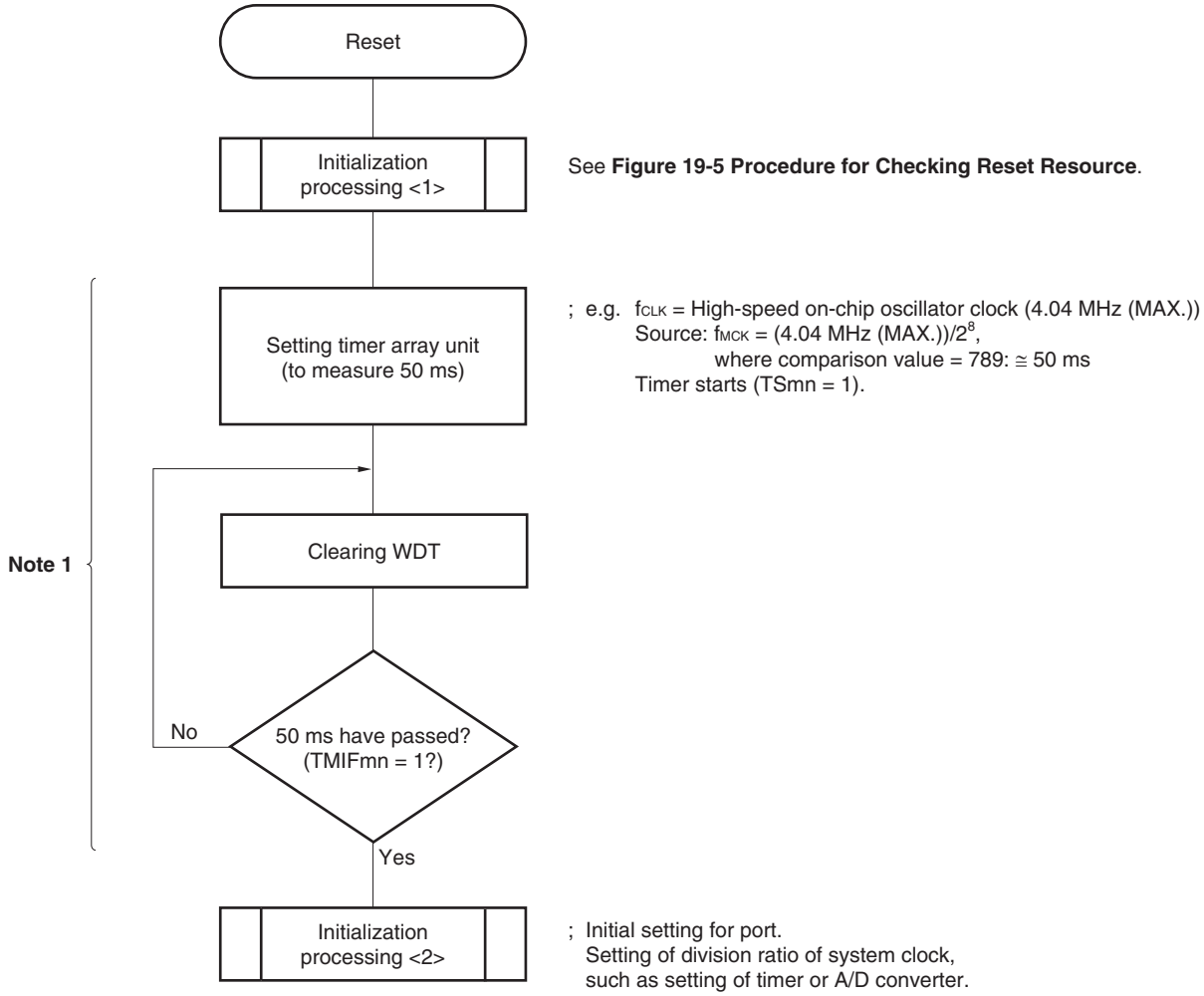
<R> **(1) Voltage fluctuation when power is supplied**

In a system where the supply voltage ($V_{DD}$) fluctuates for a certain period in the vicinity of the LVD detection voltage, the system may be repeatedly reset and released from the reset status. In this case, the time from release of reset to the start of the operation of the microcontroller can be arbitrarily set by taking the following action.

<Action>

After releasing the reset signal, wait for the supply voltage fluctuation period of each system by means of a software counter that uses a timer, and then initialize the ports.

<R> **Figure 21-9. Example of Software Processing If Supply Voltage Fluctuation Is 50 ms or Less in Vicinity of LVD Detection Voltage**




**Note** If reset is generated again during this period, initialization processing <2> is not started.

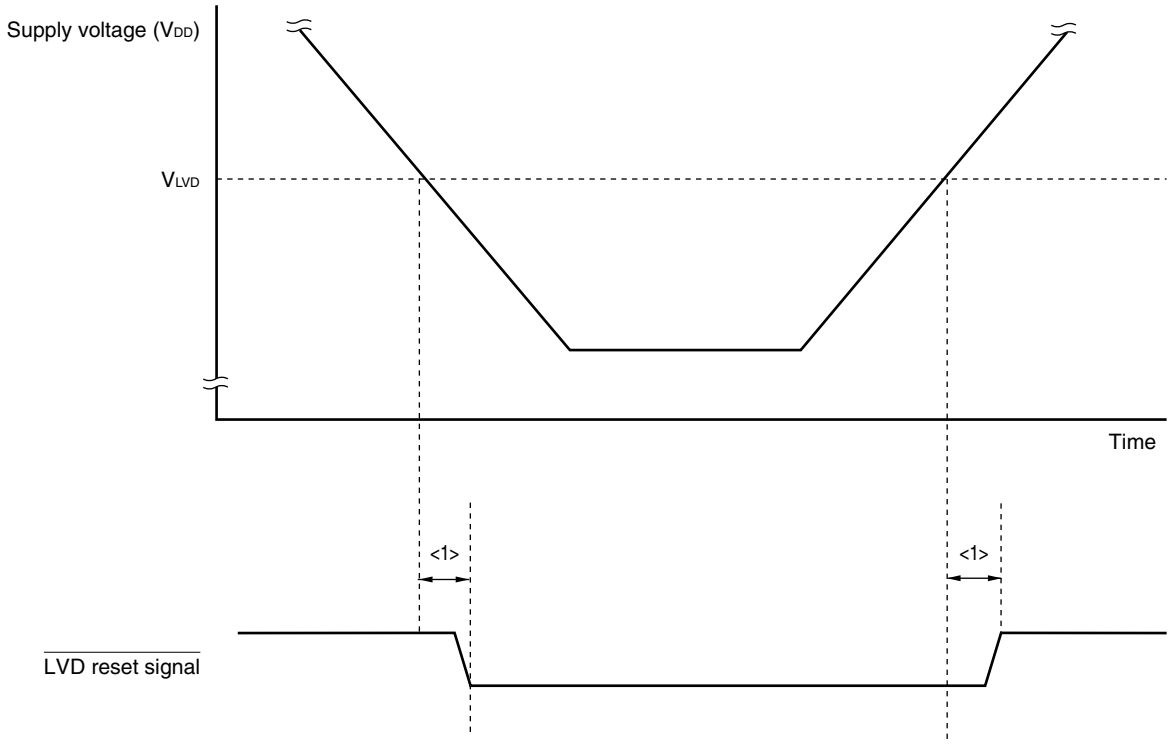
**Remark** m = 0
n = 0 to 7

**(2) Delay from the time LVD reset source is generated until the time LVD reset has been generated or released**

There is some delay from the time supply voltage ($V_{DD}$) < LVD detection voltage ($V_{LVD}$) until the time LVD reset has been generated.

In the same way, there is also some delay from the time LVD detection voltage ($V_{LVD}$) ≤ supply voltage ($V_{DD}$) until the time LVD reset has been released (see **Figure 21-10**).

**Figure 21-10. Delay from the Time LVD Reset Source Is Generated Until the Time LVD Reset has Been Generated or Released**




<1>: Detection delay (300 μs (MAX.))

<R> **(3) Power on when LVD is off**

Use the external rest input via the $\overline{\text{RESET}}$ pin when the LVD is off.

For an external reset, input a low level for 10 μs or more to the $\overline{\text{RESET}}$ pin. To perform an external reset upon power application, input a low level to the $\overline{\text{RESET}}$ pin, turn power on, continue to input a low level to the pin for 10 μs or more within the operating voltage range shown in **29.4** or **30.4 AC Characteristics**, and then input a high level to the pin.

<R> **(4) Operating voltage fall when LVD is off or LVD interrupt mode is selected**

When the operating voltage falls with the LVD is off or with the LVD interrupt mode is selected, this LSI should be placed in the STOP mode, or placed in the reset state by controlling the externally input reset signal, before the voltage falls below the operating voltage range defined in **29.4** or **30.4 AC characteristics**. When restarting the operation, make sure that the operation voltage has returned within the range of operation.

# CHAPTER 22 SAFETY FUNCTIONS

## 22.1 Overview of Safety Functions

The following safety functions are provided in the RL78/G1A to comply with the IEC60730 and IEC61508 safety standards.

These functions enable the microcontroller to self-diagnose abnormalities and stop operating if an abnormality is detected.

**(1) Flash memory CRC operation function (high-speed CRC, general-purpose CRC)**

This detects data errors in the flash memory by performing CRC operations.

Two CRC functions are provided in the RL78/G1A that can be used according to the application or purpose of use.

- High-speed CRC: The CPU can be stopped and a high-speed check executed on its entire code flash memory area during the initialization routine.
- General CRC: This can be used for checking various data in addition to the code flash memory area while the CPU is running.

**(2) RAM parity error detection function**

This detects parity errors when reading RAM data.

**(3) RAM guard function**

This prevents RAM data from being rewritten when the CPU freezes.

**(4) SFR guard function**

This prevents SFRs from being rewritten when the CPU freezes.

**(5) Invalid memory access detection function**

This detects illegal accesses to invalid memory areas (such as areas where no memory is allocated and areas to which access is restricted).

<R> **(6) Frequency detection function**

This function allows a self-check of the CPU/peripheral hardware clock frequencies using the timer array unit.

**(7) A/D test function**

This is used to perform a self-check of the A/D converter by performing A/D conversion of the A/D converter's positive and negative reference voltages, analog input channel (ANI), temperature sensor output voltage, and internal reference voltage.

**Remark** For usage examples of the safety functions complying with the IEC60730 safety standards, refer to the RL78 MCU series IEC60730/60335 application notes (R01AN1062, R01AN1296).

## 22.2 Registers Used by Safety Functions

The safety functions use the following registers for each function.

| Register | Each Function of Safety Function |
|---|---|
| • Flash memory CRC control register (CRC0CTL)<br>• Flash memory CRC operation result register (PGCRCL) | Flash memory CRC operation function (high-speed CRC) |
| • CRC input register (CRCIN)<br>• CRC data register (CRCD) | CRC operation function (general-purpose CRC) |
| • RAM parity error control register (RPECTL) | RAM parity error detection function |
| • Invalid memory access detection control register (IAWCTL) | RAM guard function |
| | SFR guard function |
| | Invalid memory access detection function |
| • Timer input select register 0 (TIS0) | Frequency detection function |
| • A/D test register (ADTES) | A/D test function |

The content of each register is described in **22.3 Operation of Safety Functions**.

## 22.3 Operation of Safety Functions

### 22.3.1 Flash memory CRC operation function (high-speed CRC)

The IEC60730 standard mandates the checking of data in the flash memory, and recommends using CRC to do it. The high-speed CRC provided in the RL78/G1A can be used to check the entire code flash memory area during the initialization routine. The high-speed CRC can be executed only when the program is allocated on the RAM and in the HALT mode of the main system clock.

The high-speed CRC performs an operation by reading 32-bit data per clock from the flash memory while stopping the CPU. This function therefore can finish a check in a shorter time (for example, 512 $\mu$s@32 MHz with 64 KB flash memory).

The CRC generator polynomial used complies with "$X^{16} + X^{12} + X^5 + 1$" of CRC-16-CCITT.

The high-speed CRC operates in MSB first order from bit 31 to bit 0.

**Caution The CRC operation result might differ during on-chip debugging because the monitor program is allocated.**

**Remark** The operation result is different between the high-speed CRC and the general CRC, because the general CRC operates in LSB first order.

#### 22.3.1.1 Flash memory CRC control register (CRC0CTL)

This register is used to control the operation of the high-speed CRC ALU, as well as to specify the operation range.

The CRC0CTL register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 22-1. Format of Flash Memory CRC Control Register (CRC0CTL)**

Address: F02F0H After reset: 00H R/W

| Symbol | <7> | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CRC0CTL | CRC0EN | 0 | FEA5 | FEA4 | FEA3 | FEA2 | FEA1 | FEA0 |

| CRC0EN | Control of CRC ALU operation |
|---|---|
| 0 | Stop the operation. |
| 1 | Start the operation according to HALT instruction execution. |

| FEA5 | FEA4 | FEA3 | FEA2 | FEA1 | FEA0 | High-speed CRC operation range |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 00000H to 03FFBH (16 Kbytes - 4 bytes) |
| 0 | 0 | 0 | 0 | 0 | 1 | 00000H to 07FFBH (32 Kbytes - 4 bytes) |
| 0 | 0 | 0 | 0 | 1 | 0 | 00000H to 0BFFBH (48 Kbytes - 4 bytes) |
| 0 | 0 | 0 | 0 | 1 | 1 | 00000H to 0FFFBH (64 Kbytes - 4 bytes) |
| Other than above | | | | | | Setting prohibited |

**Remark** Input the expected CRC operation result value to be used for comparison in the lowest 4 bytes of the flash memory. Note that the operation range will thereby be reduced by 4 bytes.

### 22.3.1.2 Flash memory CRC operation result register (PGCRCL)

This register is used to store the high-speed CRC operation results.

The PGCRCL register can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0000H.

**Figure 22-2. Format of Flash Memory CRC Operation Result Register (PGCRCL)**

Address: F02F2H After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| PGCRCL | PGCRC15 | PGCRC14 | PGCRC13 | PGCRC12 | PGCRC11 | PGCRC10 | PGCRC9 | PGCRC8 |
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PGCRC7 | PGCRC6 | PGCRC5 | PGCRC4 | PGCRC3 | PGCRC2 | PGCRC1 | PGCRC0 |

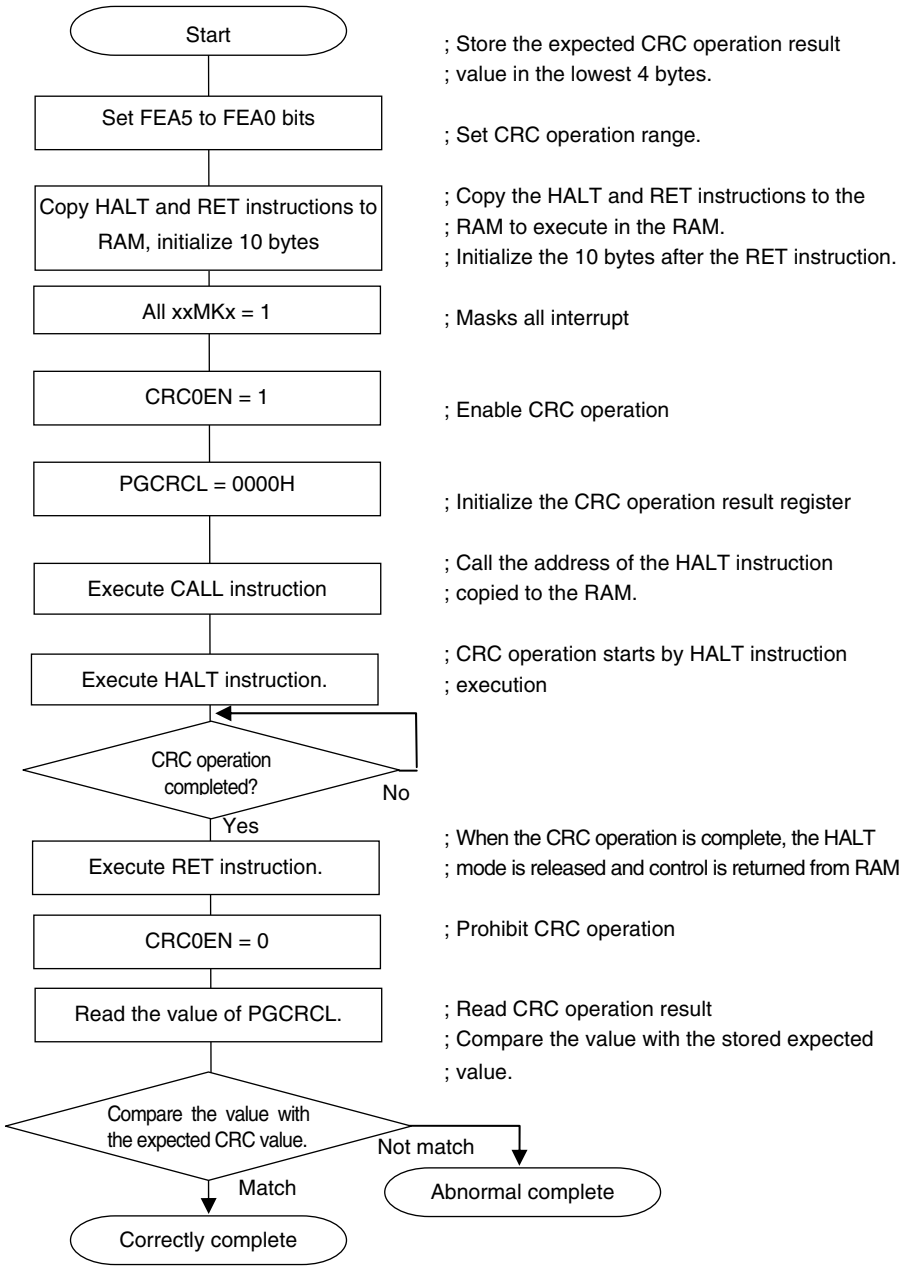
| PGCRC15 to 0 | High-speed CRC operation results |
|---|---|
| 0000H to FFFFH | Store the high-speed CRC operation results. |

**Caution The PGCRCL register can only be written if CRC0EN (bit 7 of the CRC0CTL register) = 1.**

Figure 22-3 shows the flowchart of flash memory CRC operation function (high-speed CRC).

**<Operation flow>**

<R> **Figure 22-3. Flowchart of Flash Memory CRC Operation Function (High-speed CRC)**




**Cautions 1. The CRC operation is executed only on the code flash.**

**2. Store the expected CRC operation value in the area below the operation range in the code flash.**

**3. The CRC operation is enabled by executing the HALT instruction in the RAM area.**
**Be sure to execute the HALT instruction in RAM area.**

The expected CRC value can be calculated by using the Integrated Development Environment CubeSuite+. See the Integrated Development Environment CubeSuite+ user's manual for details.

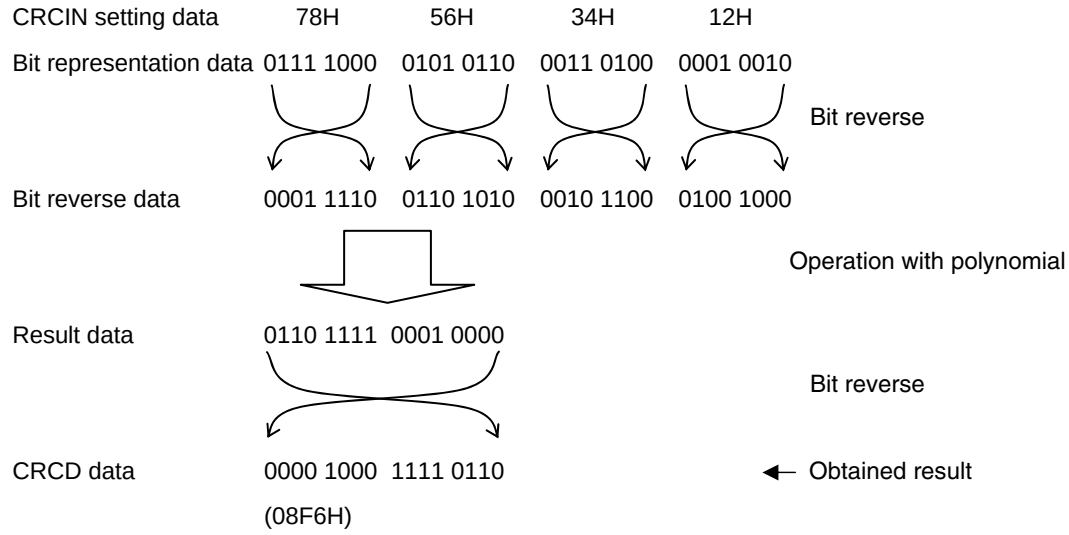
### 22.3.2 CRC operation function (general-purpose CRC)

In order to guarantee safety during operation, the IEC61508 standard mandates the checking of data even while the CPU is operating.

In the RL78/G1A, a general CRC operation can be executed as a peripheral function while the CPU is operating. The general CRC can be used for checking various data in addition to the code flash memory area. The data to be checked can be specified by using software (a user-created program). CRC calculation function in the HALT mode can be used only during the DMA transmission.

The general CRC operation can be executed in the main system clock operation mode as well as the subsystem clock operation mode.

The CRC generator polynomial used is "$X^{16} + X^{12} + X^5 + 1$" of CRC-16-CCITT. The data to be input is inverted in bit order and then calculated to allow for LSB-first communication. For example, if the data 12345678H is sent from the LSB, values are written to the CRCIN register in the order of 78H, 56H, 34H, and 12H, enabling a value of 08F6H to be obtained from the CRCD register. This is the result obtained by executing a CRC operation on the bit rows shown below, which consist of the data 12345678H inverted in bit order.




**Caution Because the debugger rewrites the software break setting line to a break instruction during program execution, the CRC operation result differs if a software break is set in the CRC operation target area.**

#### 22.3.2.1 CRC input register (CRCIN)

CRCIN register is an 8-bit register that is used to set the CRC operation data of general-purpose CRC.

The possible setting range is 00H to FFH.

The CRCIN register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 22-4. Format of CRC Input Register (CRCIN)**

Address: FFFACH After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| CRCIN | | | | | | | | |

| Bits 7 to 0 | Function |
|---|---|
| 00H to FFH | Data input. |

#### 22.3.2.2 CRC data register (CRCD)

This register is used to store the CRC operation result of the general-purpose CRC.

The setting range is 0000H to FFFFH.

After 1 clock of CPU/peripheral hardware clock ($f_{CLK}$) has elapsed from the time CRCIN register is written, the CRC operation result is stored to the CRCD register.

The CRCD register can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0000H.

**Figure 22-5. Format of CRC Data Register (CRCD)**

Address: F02FAH After reset: 0000H R/W

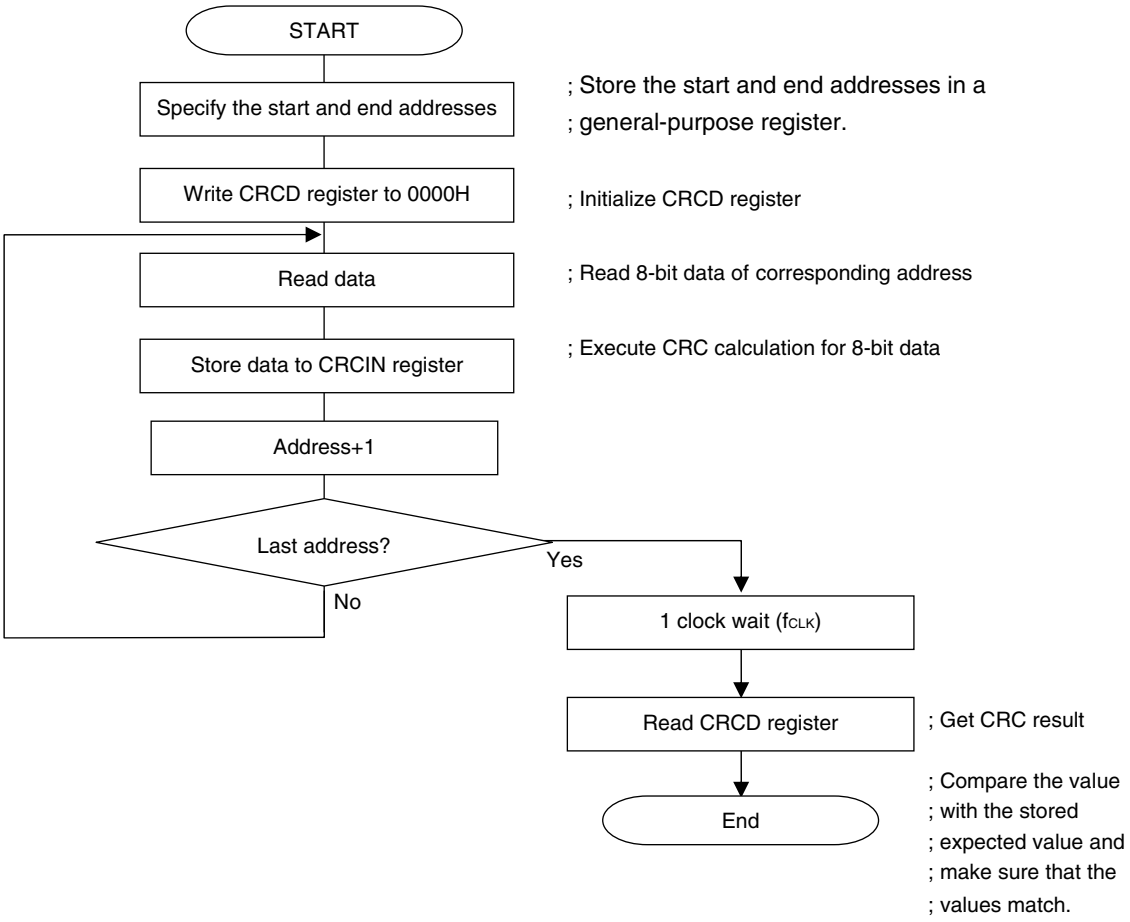
| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CRCD | | | | | | | | | | | | | | | | |

**Cautions 1. Read the value written to CRCD register before writing to CRCIN register.**

**2. If conflict between writing and storing operation result to CRCD register occurs, the writing is ignored.**

<Operation flow>

**Figure 22-6. CRC Operation Function (General-Purpose CRC)**



### 22.3.3 RAM parity error detection function

The IEC60730 standard mandates the checking of RAM data. A single-bit parity bit is therefore added to all 8-bit data in the RL78/G1A's RAM. By using this RAM parity error detection function, the parity bit is appended when data is written, and the parity is checked when the data is read. This function can also be used to trigger a reset when a parity error occurs.

<Control register>

#### 22.3.3.1 RAM parity error control register (RPECTL)

This register is used to control parity error generation check bit and reset generation due to parity errors.

The RPECTL register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 22-7. Format of RAM Parity Error Control Register (RPECTL)**

Address: F00F5H After reset: 00H R/W

| Symbol | <7> | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| RPECTL | RPERDIS | 0 | 0 | 0 | 0 | 0 | 0 | RPEF |

| RPERDIS | Parity error reset mask flag |
|---|---|
| 0 | Enable parity error resets. |
| 1 | Disable parity error resets. |

| RPEF | Parity error status flag |
|---|---|
| 0 | No parity error has occurred. |
| 1 | A parity error has occurred. |

**Caution The parity bit is appended when data is written, and the parity is checked when the data is read. Therefore, while RAM parity error resets are enabled (RPERDIS = 0), be sure to initialize RAM areas where data access is to proceed before reading data.**
**The RL78's CPU executes look-ahead due to the pipeline operation, the CPU might read an uninitialized RAM area that is allocated beyond the RAM used, which causes a RAM parity error. Therefore, while RAM parity error resets are enabled (RPERDIS = 0), be sure to initialize the RAM area + 10 bytes when instructions are fetched from RAM areas. When using the self-programming function while RAM parity error resets are enabled (RPERDIS = 0), be sure to initialize the RAM area to overwrite + 10 bytes before overwriting.**

**Remarks 1.** The parity error reset is enabled by default (RPERDIS = 0).

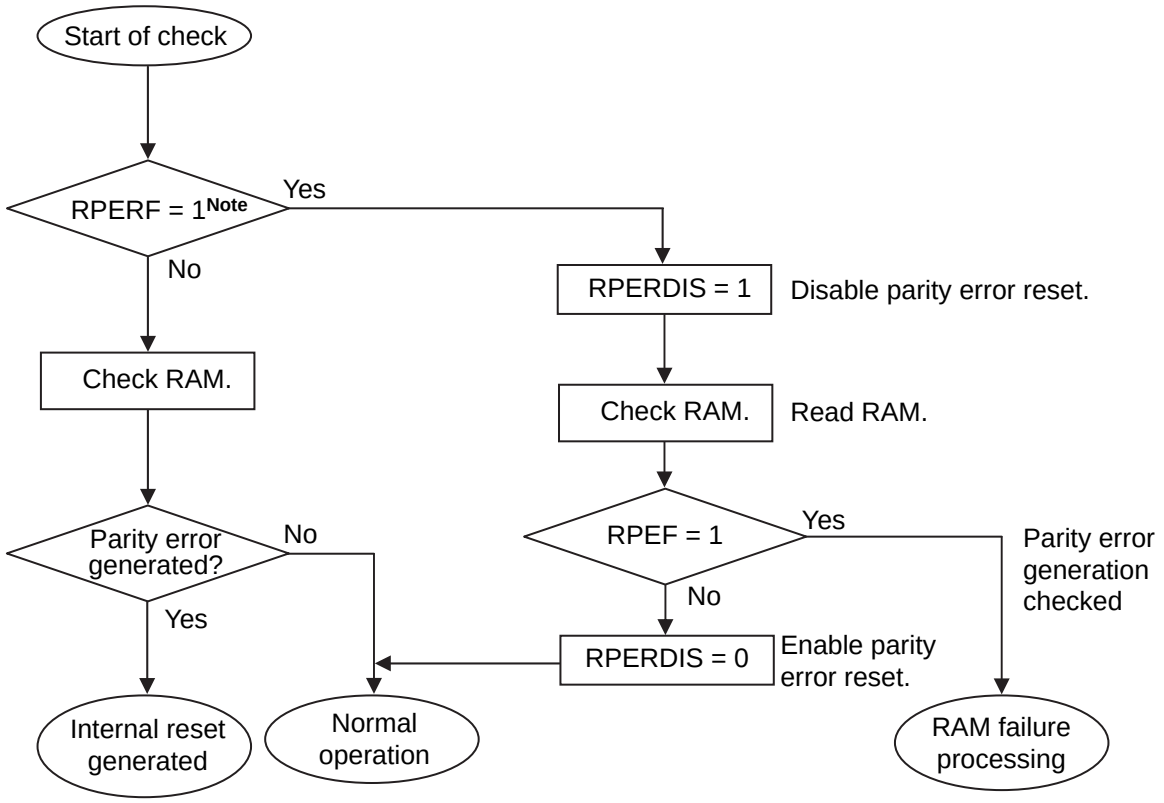
**2.** Even if the parity error reset is disabled (RPERDIS = 1), the RPEF flag will be set (1) if a parity error occurs. If parity error resets are enabled (RPERDIS = 0) with RPEF set to 1, a parity error reset is generated when the RPERDIS bit is cleared to 0.

**3.** The RPEF flag in the RPECTL register is set (1) when the RAM parity error occurs and cleared (0) by writing 0 to it or by any reset source. When RPEF = 1, the value is retained even if RAM for which no parity error has occurred is read.

**4.** The general registers are not included for RAM parity error detection.

<R>

**Figure 22-8. Flowchart of RAM Parity Check**




**Note** To check internal reset status using a RAM parity error, see **CHAPTER 19 RESET FUNCTION**.

### 22.3.4 RAM guard function

In order to guarantee safety during operation, the IEC61508 standard mandates that important data stored in the RAM be protected, even if the CPU freezes.

This RAM guard function is used to protect data in the specified memory space.

If the RAM guard function is specified, writing to the specified RAM space is disabled, but reading from the space can be carried out as usual.

#### 22.3.4.1 Invalid memory access detection control register (IAWCTL)

This register is used to control the detection of invalid memory access and RAM/SFR guard function.

GRAM1 and GRAM0 bits are used in RAM guard function.

The IAWCTL register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 22-9. Format of Invalid Memory Access Detection Control Register (IAWCTL)**

Address: F0078H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IAWCTL | IAWEN | 0 | GRAM1 | GRAM0 | 0 | GPORT | GINT | GCSC |

| | GRAM1 | GRAM0 | RAM guard space[Note] |
|---|---|---|---|
| | 0 | 0 | Disabled. RAM can be written to. |
| <R> | 0 | 1 | The 128 bytes of space starting at the start address in the RAM |
| <R> | 1 | 0 | The 256 bytes of space starting at the start address in the RAM |
| <R> | 1 | 1 | The 512 bytes of space starting at the start address in the RAM |

**Note** The RAM start address differs depending on the size of the RAM provided with the product.

### 22.3.5 SFR guard function

In order to guarantee safety during operation, the IEC61508 standard mandates that important SFRs be protected from being overwritten, even if the CPU freezes.

This SFR guard function is used to protect data in the control registers used by the port function, interrupt function, clock control function, voltage detection function, and RAM parity error function.

If the SFR guard function is specified, writing to the specified SFRs is disabled, but reading from the SFRs can be carried out as usual.

#### 22.3.5.1 Invalid memory access detection control register (IAWCTL)

This register is used to control the detection of invalid memory access and RAM/SFR guard function.

GPORT, GINT and GCSC bits are used in SFR guard function.

The IAWCTL register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 22-10. Format of Invalid Memory Access Detection Control Register (IAWCTL)**

Address: F0078H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IAWCTL | IAWEN | 0 | GRAM1 | GRAM0 | 0 | GPORT | GINT | GCSC |

| GPORT | Control registers of port function guard |
|---|---|
| 0 | Disabled. Control registers of port function can be read or written to. |
| 1 | Enabled. Writing to control registers of port function is disabled. Reading is enabled.<br>[Guarded SFR] PMxx, PUxx, PIMxx, POMxx, PMCxx, ADPC, PIOR[Note] |

| GINT | Registers of interrupt function guard |
|---|---|
| 0 | Disabled. Registers of interrupt function can be read or written to. |
| 1 | Enabled. Writing to registers of interrupt function is disabled. Reading is enabled.<br>[Guarded SFR] IFxx, MKxx, PRxx, EGPx, EGNx |

| GCSC | Control registers of clock control function, voltage detector and RAM parity error detection function guard |
|---|---|
| 0 | Disabled. Control registers of clock control function, voltage detector and RAM parity error detection function can be read or written to. |
| 1 | Enabled. Writing to control registers of clock control function, voltage detector and RAM parity error detection function is disabled. Reading is enabled.<br>[Guarded SFR] CMC, CSC, OSTS, CKC, PERx, OSMC, LVIM, LVIS, RPECTL |

**Note** Pxx (Port register) is not guarded.

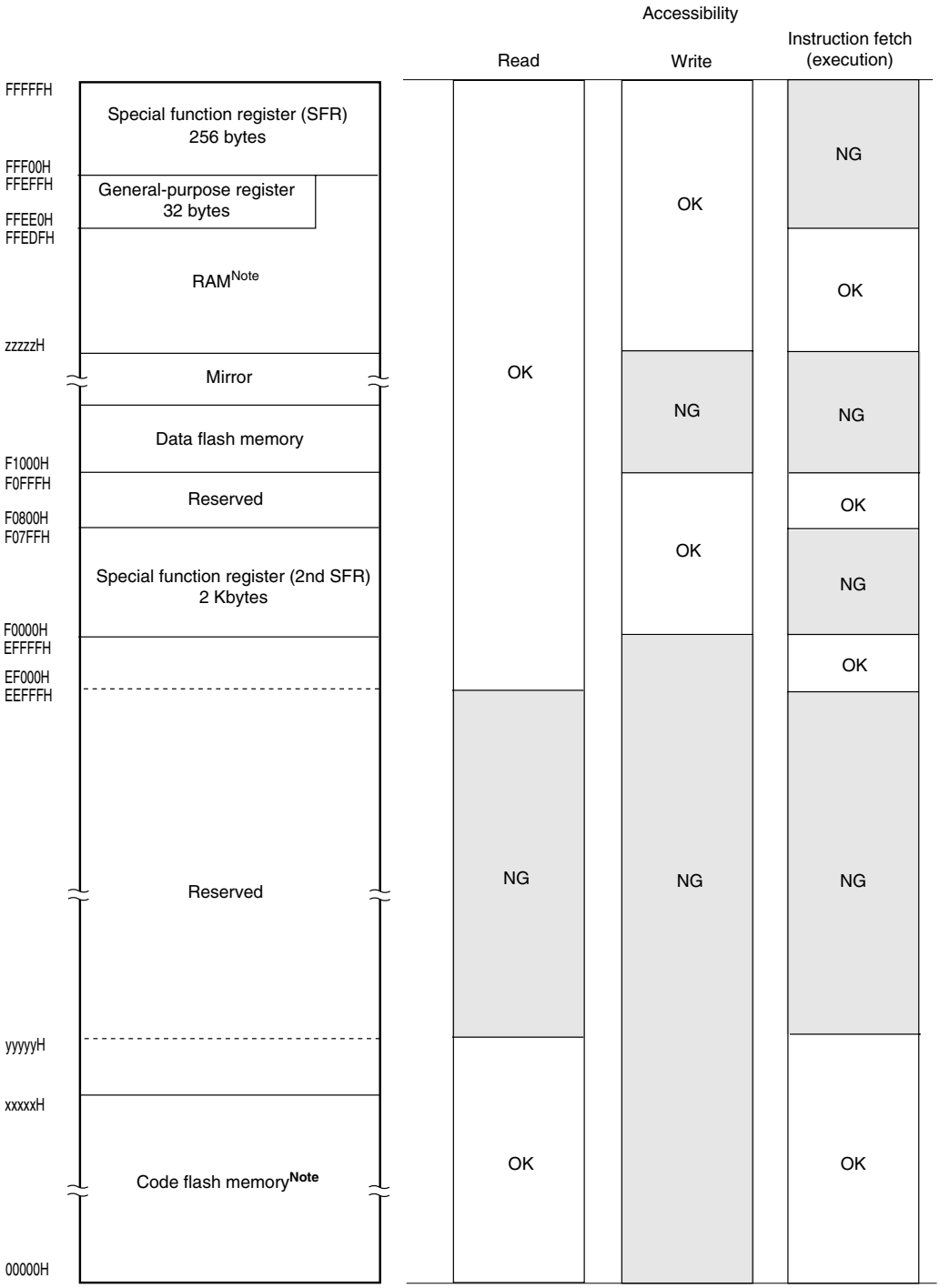
### 22.3.6 Invalid memory access detection function

The IEC60730 standard mandates checking that the CPU and interrupts are operating correctly.

The illegal memory access detection function triggers a reset if a memory space specified as access-prohibited is accessed.

The illegal memory access detection function applies to the areas indicated by NG in Figure 22-11.

<R> **Figure 22-11. Invalid Access Detection Area**




**Note** The following table lists the code flash memory, RAM, and lowest detection address for each product:

(The explanation is described on the next page.)

| Products | Code Flash Memory (00000H to xxxxxH) | RAM (zzzzzH to FFEFFH) | Detected Lowest Address for Read/Instruction Fetch (Execution) (yyyyyH) |
|---|---|---|---|
| R5F10ExA (x = 8, B, G) | 16384 × 8 bit (00000H to 03FFFH) | 2048 × 8 bit (FF700H to FFEFFH) | 10000H |
| R5F10ExC (x = 8, B, G, ,L) | 32768 × 8 bit (00000H to 07FFFH) | 2048 × 8 bit (FF700H to FFEFFH) | 10000H |
| R5F10ExD (x = 8, B, G, L) | 49152 × 8 bit (00000H to 0BFFFH) | 3072 × 8 bit (FF300H to FFEFFH) | 10000H |
| R5F10ExE (x = 8, B, G, L) | 65536 × 8 bit (00000H to 0FFFFH) | 4096 × 8 bit (FEF00H to FFEFFH) | 10000H |

#### 22.3.6.1 Invalid memory access detection control register (IAWCTL)

This register is used to control the detection of invalid memory access and RAM/SFR guard function.

IAWEN bit is used in invalid memory access detection function.

The IAWCTL register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 22-12. Format of Invalid Memory Access Detection Control Register (IAWCTL)**

Address: F0078H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| IAWCTL | IAWEN | 0 | GRAM1 | GRAM0 | 0 | GPORT | GINT | GCSC |

| IAWEN[Note] | Control of invalid memory access detection |
|---|---|
| 0 | Disable the detection of invalid memory access. |
| 1 | Enable the detection of invalid memory access. |

**Note** Only writing 1 to the IAWEN bit is enabled, not writing 0 to it after setting it to 1.

**Remark** By specifying WDTON = 1 (watchdog timer operation enable) for the option byte (000C0H), the invalid memory access function is enabled even IAWEN = 0.

### 22.3.7 Frequency detection function

The IEC60730 standard mandates checking that the oscillation frequency is correct.

<R> By using the CPU/peripheral hardware clock frequency ($f_{CLK}$) and measuring the pulse width of the input signal to channel 5 of the timer array unit 0 (TAU0), whether the proportional relationship between the two clock frequencies is correct can be determined. Note that, however, if one or both clock operations are completely stopped, the proportional relationship between the clocks cannot be determined.

<Clocks to be compared>

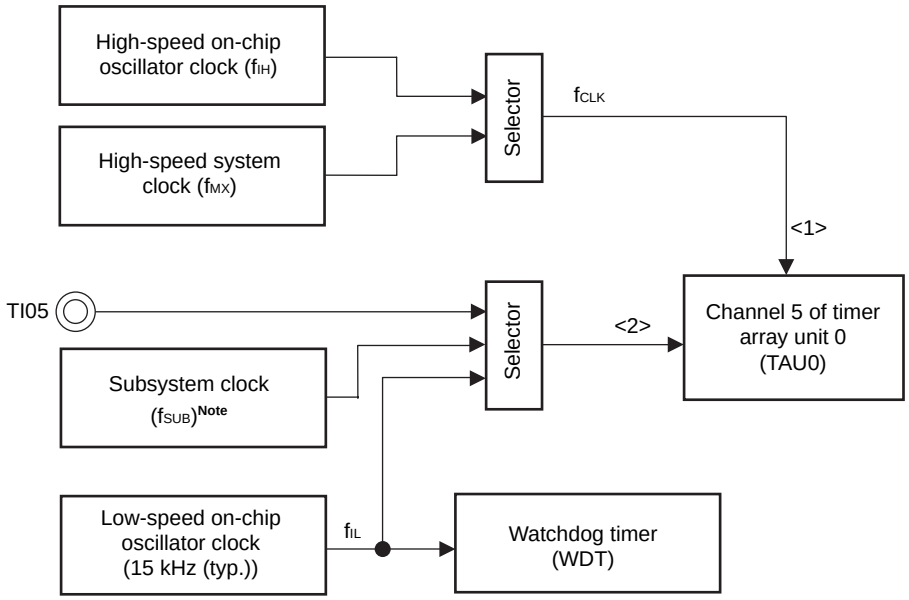
<1> CPU/peripheral hardware clock frequency ($f_{CLK}$):

- High-speed on-chip oscillator clock ($f_{IH}$)
- High-speed system clock ($f_{MX}$)

<2> Input to channel 5 of the timer array unit

- Timer input to channel 5 (TI05)
- Low-speed on-chip oscillator clock ($f_{IL}$: 15 kHz (typ.))
- Subsystem clock ($f_{SUB}$)Note

**Figure 22-13. Configuration of Frequency Detection Function**




If input pulse interval measurement results in an abnormal value, it can be concluded that the clock frequency is abnormal.

For how to execute input pulse interval measurement, see **6.8.4 Operation as input pulse interval measurement**.

**Note** Can only be selected in the products incorporating the subsystem clock.

#### 22.3.7.1 Timer input select register 0 (TIS0)

<R> The TIS0 register is used to select the timer input of channel 5 of the timer array unit 0 (TAU0).

The TIS0 register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 22-14. Format of Timer Input Select Register 0 (TIS0)**

Address: F0074H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| TIS0 | 0 | 0 | 0 | 0 | 0 | TIS02 | TIS01 | TIS00 |

| TIS02 | TIS01 | TIS00 | Selection of timer input used with channel 5 |
|---|---|---|---|
| 0 | 0 | 0 | Input signal of timer input pin (TI05) |
| 0 | 0 | 1 | |
| 0 | 1 | 0 | |
| 0 | 1 | 1 | |
| 1 | 0 | 0 | Low-speed on-chip oscillator clock ($f_{IL}$) |
| 1 | 0 | 1 | Subsystem clock ($f_{SUB}$) |
| Other than above | | | Setting prohibited |

### 22.3.8 A/D test function

<R> The IEC60730 standard mandates testing the A/D converter. The A/D test function checks whether or not the A/D converter is operating normally by executing A/D conversions of the A/D converter's positive and negative reference voltages, analog input channel (ANI), temperature sensor output voltage, and the internal reference voltage. For details of the check method, see the safety function (A/D test) application note (R01AN0955).

The analog multiplexer can be checked using the following procedure.

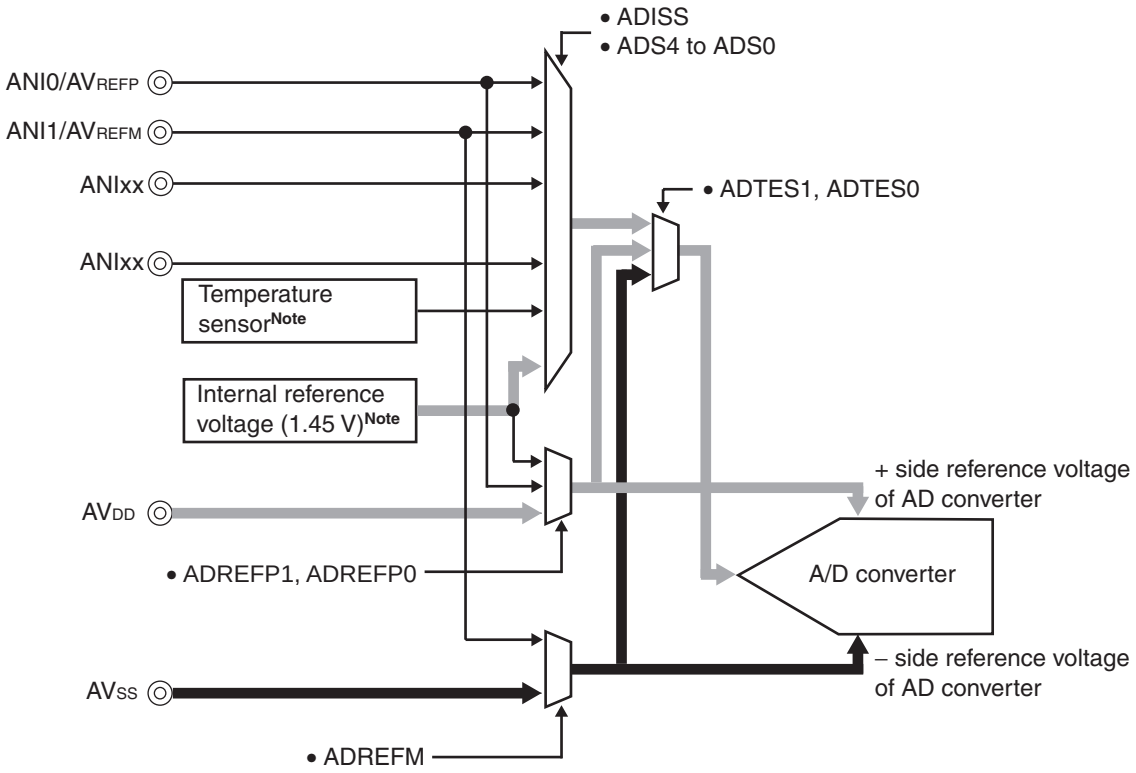
<1> Select the ANIx pin for A/D conversion using the ADTES register (ADTES1 = 0, ADTES0 = 0).
<2> Perform A/D conversion for the ANIx pin (conversion result 1-1).
<3> Select the A/D converter's negative reference voltage for A/D conversion using the ADTES register (ADTES1 = 1, ADTES0 = 0)
<4> Perform A/D conversion of the negative reference voltage of the A/D converter (conversion result 2-1).
<5> Select the ANIx pin for A/D conversion using the ADTES register (ADTES1 = 0, ADTES0 = 0).
<6> Perform A/D conversion for the ANIx pin (conversion result 1-2).
<7> Select the A/D converter's positive reference voltage for A/D conversion using the ADTES register (ADTES1 = 1, ADTES0 = 1)
<8> Perform A/D conversion of the positive reference voltage of the A/D converter (conversion result 2-2).
<9> Select the ANIx pin for A/D conversion using the ADTES register (ADTES1 = 0, ADTES0 = 0).
<10> Perform A/D conversion for the ANIx pin (conversion result 1-3).
<11> Check that the conversion results 1-1, 1-2, and 1-3 are equal.
<12> Check that the A/D conversion result 2-1 is all zero and conversion result 2-2 is all one.

Using the procedure above can confirm that the analog multiplexer is selected and all wiring is connected.

**Remarks 1.** If the analog input voltage is variable during A/D conversion in steps <1> to <10> above, use another method to check the analog multiplexer.

**2.** The conversion results might contain an error. Consider an appropriate level of error when comparing the conversion results.

<R> **Figure 22-15. Configuration of A/D Test Function**




**Note** This setting can be used only in HS (high-speed main) mode.

#### 22.3.8.1 A/D test register (ADTES)

<R> This register is used to select the A/D converter's positive reference voltage, A/D converter's negative reference voltage, analog input channel (ANIxx), temperature sensor output voltage, or internal reference voltage (1.45 V) as the target of A/D conversion.

When using the A/D test function, specify the following settings:

- Select negative reference voltage as the target of A/D conversion for zero-scale measurement.
- Select positive reference voltage as the target of A/D conversion for full-scale measurement.

The ADTES register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 22-16. Format of A/D Test Register (ADTES)**

Address: F0013H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADTES | 0 | 0 | 0 | 0 | 0 | 0 | ADTES1 | ADTES0 |

| ADTES1 | ADTES0 | A/D conversion target |
|---|---|---|
| 0 | 0 | ANIxx/temperature sensor output voltage[Note]/internal reference voltage (1.45 V)[Note] (This is specified using the analog input channel specification register (ADS).) |
| 1 | 0 | Negative reference voltage (selected with the ADREFM bit in ADM2) |
| 1 | 1 | Positive reference voltage (selected with the ADREFP1 or ADREFP0 bit in ADM2) |
| Other than above | | Setting prohibited |

**Note** Temperature sensor output voltage and internal reference voltage (1.45 V) can be used only in HS (high-speed main) mode.

#### 22.3.8.2 Analog input channel specification register (ADS)

This register specifies the input channel of the analog voltage to be A/D converted.

Set A/D test register (ADTES) to 00H when measuring the ANIxx/temperature sensor output voltage/internal reference voltage (1.45 V).

The ADS register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 22-17. Format of Analog Input Channel Specification Register (ADS) (1/2)**

Address: FFF31H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| ADS | ADISS | 0 | 0 | ADS4 | ADS3 | ADS2 | ADS1 | ADS0 |

○ Select mode (ADMD = 0)

| ADISS | ADS4 | ADS3 | ADS2 | ADS1 | ADS0 | Analog input channel | Input source |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | ANI0 | P20/ANI0/$AV_{REFP}$ pin |
| 0 | 0 | 0 | 0 | 0 | 1 | ANI1 | P21/ANI1/$AV_{REFM}$ pin |
| 0 | 0 | 0 | 0 | 1 | 0 | ANI2 | P22/ANI2 pin |
| 0 | 0 | 0 | 0 | 1 | 1 | ANI3 | P23/ANI3 pin |
| 0 | 0 | 0 | 1 | 0 | 0 | ANI4 | P24/ANI4 pin |
| 0 | 0 | 0 | 1 | 0 | 1 | ANI5 | P25/ANI5 pin |
| 0 | 0 | 0 | 1 | 1 | 0 | ANI6 | P26/ANI6 pin |
| 0 | 0 | 0 | 1 | 1 | 1 | ANI7 | P27/ANI7 pin |
| 0 | 0 | 1 | 0 | 0 | 0 | ANI8 | P150/ANI8 pin |
| 0 | 0 | 1 | 0 | 0 | 1 | ANI9 | P151/ANI9 pin |
| 0 | 0 | 1 | 0 | 1 | 0 | ANI10 | P152/ANI10 pin |
| 0 | 0 | 1 | 0 | 1 | 1 | ANI11 | P153/ANI11 pin |
| 0 | 0 | 1 | 1 | 0 | 0 | ANI12 | P154/ANI12 pin |
| 0 | 0 | 1 | 1 | 0 | 1 | Setting prohibited | |
| 0 | 0 | 1 | 1 | 1 | 0 | Setting prohibited | |
| 0 | 0 | 1 | 1 | 1 | 1 | Setting prohibited | |
| 0 | 1 | 0 | 0 | 0 | 0 | ANI16 | P03/ANI16 pin |
| 0 | 1 | 0 | 0 | 0 | 1 | ANI17 | P02/ANI17 pin |
| 0 | 1 | 0 | 0 | 1 | 0 | ANI18 | P10/ANI18 pin |
| 0 | 1 | 0 | 0 | 1 | 1 | ANI19 | P120/ANI19 pin |
| 0 | 1 | 0 | 1 | 0 | 0 | ANI20 | P11/ANI20 pin |
| 0 | 1 | 0 | 1 | 0 | 1 | ANI21 | P12/ANI21 pin |
| 0 | 1 | 0 | 1 | 1 | 0 | ANI22 | P13/ANI22 pin |
| 0 | 1 | 0 | 1 | 1 | 1 | ANI23 | P14/ANI23 pin |
| 0 | 1 | 1 | 0 | 0 | 0 | ANI24 | P15/ANI24 pin |
| 0 | 1 | 1 | 0 | 0 | 1 | ANI25 | P51/ANI25 pin |
| 0 | 1 | 1 | 0 | 1 | 0 | ANI26 | P50/ANI26 pin |
| 0 | 1 | 1 | 0 | 1 | 1 | ANI27 | P30/ANI27 pin |
| 0 | 1 | 1 | 1 | 0 | 0 | ANI28 | P70/ANI28 pin |
| 0 | 1 | 1 | 1 | 0 | 1 | ANI29 | P31/ANI29 pin |
| 0 | 1 | 1 | 1 | 1 | 0 | ANI30 | P41/ANI30 pin |
| 0 | 1 | 1 | 1 | 1 | 1 | Setting prohibited | |
| 1 | 0 | 0 | 0 | 0 | 0 | – | Temperature sensor output voltage[Note] |
| 1 | 0 | 0 | 0 | 0 | 1 | – | Internal reference voltage output (1.45 V)[Note] |
| Other than above | | | | | | Setting prohibited | |

**Note** This setting can be used only in HS (high-speed main) mode. For details, see **Figure 24-3**.

**(Cautions 1** to **9** are listed on the next page.)

**Cautions 1. Be sure to clear bits 5 and 6 to 0.**

**2. Select input mode for the ports which are set to analog input with the ADPC and PMC registers, using the port mode registers 0 to 5, 7, 12, and 15 (PM0 to PM5, PM7, PM12, and PM15).**

**3. Do not use the ADS register to set the pins which should be set as digital I/O with the A/D port configuration register (ADPC).**

**4. Do not use the ADS register to set the pins which should be set as digital I/O with the port mode control registers 0, 1, 3 to 5, 7, and 12 (PMC0, PMC1, PMC3 to PMC5, PMC7, and PMC12).**

**5. Only rewrite the value of the ADISS bit while conversion operation is stopped (ADCS = 0, ADCE = 0).**

**6. If using $AV_{REFP}$ as the positive reference voltage source of the A/D converter, do not select ANI0 as an A/D conversion channel.**

**7. If using $AV_{REFM}$ as the negative reference voltage source of the A/D converter, do not select ANI1 as an A/D conversion channel.**

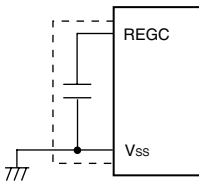
**8. When ADISS is 1, the internal reference voltage (1.45 V) cannot be used for the positive reference voltage. In addition, the first conversion result obtained after setting ADISS to 1 is not available. For detailed setting flow, see 11.7.4 Setup when temperature sensor output voltage/internal reference voltage is selected (example for software trigger mode and one-shot conversion mode).**

**9. If a transition is made to STOP mode or a transition is made to HALT mode during CPU operation with subsystem clock, do not set ADISS to 1. When ADISS is 1, the A/D converter reference voltage current ($I_{ADREF}$) shown in 29.3.2 Supply current characteristics is added.**

# CHAPTER 23 REGULATOR

## 23.1 Regulator Overview

The RL78/G1A contains a circuit for operating the device with a constant voltage. At this time, in order to stabilize the regulator output voltage, connect the REGC pin to $V_{SS}$ via a capacitor (0.47 to 1 $\mu$F). Also, use a capacitor with good characteristics, since it is used to stabilize internal voltage.




**Caution Keep the wiring length as short as possible for the broken-line part in the above figure.**

The regulator output voltage, see **Table 23-1**.

**Table 23-1. Regulator Output Voltage Conditions**

| Mode | Output Voltage | Condition |
|---|---|---|
| LV (Low voltage main) mode | 1.8 V | – |
| LS (Low-speed main) mode | | |
| HS (High-speed main) mode | 1.8 V | In STOP mode |
| | | When both the high-speed system clock ($f_{MX}$) and the high-speed on-chip oscillator clock ($f_{IH}$) are stopped during CPU operation with the subsystem clock ($f_{SUB}$) |
| | | When both the high-speed system clock ($f_{MX}$) and the high-speed on-chip oscillator clock ($f_{IH}$) are stopped during the HALT mode when the CPU operation with the subsystem clock ($f_{SUB}$) has been set |
| | 2.1 V | Other than above (include during OCD mode)[Note] |

**Note** When it shifts to the subsystem clock operation or STOP mode during the on-chip debugging, the regulator output voltage is kept at 2.1 V (not decline to 1.8 V).

# CHAPTER 24 OPTION BYTE

## 24.1 Functions of Option Bytes

Addresses 000C0H to 000C3H of the flash memory of the RL78/G1A form an option byte area.

Option bytes consist of user option byte (000C0H to 000C2H) and on-chip debug option byte (000C3H).

Upon power application or resetting and starting, an option byte is automatically referenced and a specified function is set.

For the bits to which no function is allocated, do not change their initial values.

To use the boot swap operation during self programming, 000C0H to 000C3H are replaced by 010C0H to 010C3H. Therefore, set the same values as 000C0H to 000C3H to 010C0H to 010C3H.

**Remark** The option bytes should always be set regardless of whether each function is used.

### 24.1.1 User option byte (000C0H to 000C2H/010C0H to 010C2H)

<R> (1) 000C0H/010C0H

- ○ Setting of watchdog timer operation
  - Enabling or disabling of counter operation
  - Enabling or disabling of counter operation in the HALT or STOP mode
- ○ Setting of overflow time of watchdog timer
- ○ Setting of window open period of watchdog timer
- ○ Setting of interval interrupt of watchdog timer
  - Whether or not to use the interval interrupt is selectable.

**Caution Set the same value as 000C0H to 010C0H when the boot swap operation is used because 000C0H is replaced by 010C0H.**

(2) 000C1H/010C1H

- ○ Setting of LVD operation mode
  - Interrupt & reset mode.
  - Reset mode.
  - Interrupt mode.
- <R> • LVD off (by controlling the externally input reset signal on the $\overline{\text{RESET}}$ pin)
- ○ Setting of LVD detection level ($V_{LVDH}$, $V_{LVDL}$, $V_{LVD}$)

**Cautions 1. After power is supplied, the reset state must be retained until the operating voltage becomes in the range defined in 29.4 or 30.4 AC Characteristics. This is done by utilizing the voltage detection circuit or controlling the externally input reset signal. After the power supply is turned off, this LSI should be placed in the STOP mode, or placed in the reset state by utilizing the voltage detection circuit or controlling the externally input reset signal, before the voltage falls below the operating range. The range of operating voltage varies with the setting of the user option byte (000C2H or 010C2H).**

**2. Set the same value as 000C1H to 010C1H when the boot swap operation is used because 000C1H is replaced by 010C1H.**

(3) 000C2H/010C2H

○ Setting of flash operation mode
- LV (low voltage main) mode
- LS (low speed main) mode
- HS (high speed main) mode

○ Setting of the frequency of the high-speed on-chip oscillator

<R>
- Select from 32 MHz/24 MHz/16 MHz/12 MHz/8 MHz/6 MHz/4 MHz/3 MHz/2 MHz/1 MHz (TYP.).

**Caution Set the same value as 000C2H to 010C2H when the boot swap operation is used because 000C2H is replaced by 010C2H.**

### 24.1.2 On-chip debug option byte (000C3H/ 010C3H)

○ Control of on-chip debug operation
- On-chip debug operation is disabled or enabled.

○ Handling of data of flash memory in case of failure in on-chip debug security ID authentication
- Data of flash memory is erased or not erased in case of failure in on-chip debug security ID authentication.

**Caution Set the same value as 000C3H to 010C3H when the boot swap operation is used because 000C3H is replaced by 010C3H.**

## 24.2 Format of User Option Byte

The format of user option byte is shown below.

**Figure 24-1. Format of User Option Byte (000C0H/010C0H)**

Address: 000C0H/010C0H[Note 1]

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| WDTINIT | WINDOW1 | WINDOW0 | WDTON | WDCS2 | WDCS1 | WDCS0 | WDSTBYON |

| WDTINIT | Use of interval interrupt of watchdog timer |
|---|---|
| 0 | Interval interrupt is not used. |
| 1 | Interval interrupt is generated when 75% + 1/2$f_{IL}$ of the overflow time is reached. |

| WINDOW1 | WINDOW0 | Watchdog timer window open period[Note 2] |
|---|---|---|
| 0 | 0 | Setting prohibited |
| 0 | 1 | 50% |
| 1 | 0 | 75% |
| 1 | 1 | 100% |

| WDTON | Operation control of watchdog timer counter |
|---|---|
| 0 | Counter operation disabled (counting stopped after reset) |
| 1 | Counter operation enabled (counting started after reset) |

| WDCS2 | WDCS1 | WDCS0 | Watchdog timer overflow time ($f_{IL}$ = 17.25 kHz (MAX.)) |
|---|---|---|---|
| 0 | 0 | 0 | $2^6/f_{IL}$ (3.71 ms) |
| 0 | 0 | 1 | $2^7/f_{IL}$ (7.42 ms) |
| 0 | 1 | 0 | $2^8/f_{IL}$ (14.84 ms) |
| 0 | 1 | 1 | $2^9/f_{IL}$ (29.68 ms) |
| 1 | 0 | 0 | $2^{11}/f_{IL}$ (118.72 ms) |
| 1 | 0 | 1 | $2^{13}/f_{IL}$ (474.89 ms) |
| 1 | 1 | 0 | $2^{14}/f_{IL}$ (949.79 ms) |
| 1 | 1 | 1 | $2^{16}/f_{IL}$ (3799.18 ms) |

| WDSTBYON | Operation control of watchdog timer counter (HALT/STOP mode) |
|---|---|
| 0 | Counter operation stopped in HALT/STOP mode[Note 2] |
| 1 | Counter operation enabled in HALT/STOP mode |

**Notes 1.** Set the same value as 000C0H to 010C0H when the boot swap operation is used because 000C0H is replaced by 010C0H.

**2.** The window open period is 100% when WDSTBYON = 0, regardless the value of the WINDOW1 and WINDOW0 bits.

**Remark** $f_{IL}$: Low-speed on-chip oscillator clock frequency

**Figure 24-2. Format of User Option Byte (000C1H/010C1H) (1/2)**

Address: 000C1H/010C1H[Note]

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| VPOC2 | VPOC1 | VPOC0 | 1 | LVIS1 | LVIS0 | LVIMDS1 | LVIMDS0 |

<R> • LVD setting (interrupt & reset mode)

| Detection voltage | | | Option byte setting value | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $V_{LVDH}$ | | $V_{LVDL}$ | VPOC2 | VPOC1 | VPOC0 | LVIS1 | LVIS0 | Mode setting | |
| Rising edge | Falling edge | Falling edge | | | | | | LVIMDS1 | LVIMDS0 |
| 1.77 V | 1.73 V | 1.63 V | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
| 1.88 V | 1.84 V | | | | | 0 | 1 | | |
| 2.92 V | 2.86 V | | | | | 0 | 0 | | |
| 1.98 V | 1.94 V | 1.84 V | | 0 | 1 | 1 | 0 | | |
| 2.09 V | 2.04 V | | | | | 0 | 1 | | |
| 3.13 V | 3.06 V | | | | | 0 | 0 | | |
| 2.61 V | 2.55 V | 2.45 V | | 1 | 0 | 1 | 0 | | |
| 2.71 V | 2.65 V | | | | | 0 | 1 | | |
| 2.92 V | 2.86 V | 2.75 V | | 1 | 1 | 1 | 0 | | |
| 3.02 V | 2.96 V | | | | | 0 | 1 | | |
| – | | | Setting of values other than above is prohibited. | | | | | | |

<R> • LVD setting (reset mode)

| Detection voltage | | Option byte setting value | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $V_{LVD}$ | | VPOC2 | VPOC1 | VPOC0 | LVIS1 | LVIS0 | Mode setting | |
| Rising edge | Falling edge | | | | | | LVIMDS1 | LVIMDS0 |
| 1.67 V | 1.63 V | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 1.77 V | 1.73 V | | 0 | 0 | 1 | 0 | | |
| 1.88 V | 1.84 V | | 0 | 1 | 1 | 1 | | |
| 1.98 V | 1.94 V | | 0 | 1 | 1 | 0 | | |
| 2.09 V | 2.04 V | | 0 | 1 | 0 | 1 | | |
| 2.50 V | 2.45 V | | 1 | 0 | 1 | 1 | | |
| 2.61 V | 2.55 V | | 1 | 0 | 1 | 0 | | |
| 2.71 V | 2.65 V | | 1 | 0 | 0 | 1 | | |
| 2.81 V | 2.75 V | | 1 | 1 | 1 | 1 | | |
| 2.92 V | 2.86 V | | 1 | 1 | 1 | 0 | | |
| 3.02 V | 2.96 V | | 1 | 1 | 0 | 1 | | |
| 3.13 V | 3.06 V | | 0 | 1 | 0 | 0 | | |
| – | | Setting of values other than above is prohibited. | | | | | | |

**Note** Set the same value as 000C1H to 010C1H when the boot swap operation is used because 000C1H is replaced by 010C1H.

**Remarks 1.** For details on the LVD circuit, see **CHAPTER 21 VOLTAGE DETECTOR.**

**2.** The detection voltage is a typical value. For details, see **29.6.4** or **30.6.4 LVD circuit characteristics.**

(**Cautions** are listed on the next page.)

**Figure 24-2. Format of User Option Byte (000C1H/010C1H) (2/2)**

Address: 000C1H/010C1H[Note]

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| VPOC2 | VPOC1 | VPOC0 | 1 | LVIS1 | LVIS0 | LVIMDS1 | LVIMDS0 |

<R> • LVD setting (interrupt mode)

| Detection voltage | | Option byte setting value | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $V_{LVD}$ | | VPOC2 | VPOC1 | VPOC0 | LVIS1 | LVIS0 | Mode setting | |
| Rising edge | Falling edge | | | | | | LVIMDS1 | LVIMDS0 |
| 1.67 V | 1.63 V | 0 | 0 | 0 | 1 | 1 | 0 | 1 |
| 1.77 V | 1.73 V | | 0 | 0 | 1 | 0 | | |
| 1.88 V | 1.84 V | | 0 | 1 | 1 | 1 | | |
| 1.98 V | 1.94 V | | 0 | 1 | 1 | 0 | | |
| 2.09 V | 2.04 V | | 0 | 1 | 0 | 1 | | |
| 2.50 V | 2.45 V | | 1 | 0 | 1 | 1 | | |
| 2.61 V | 2.55 V | | 1 | 0 | 1 | 0 | | |
| 2.71 V | 2.65 V | | 1 | 0 | 0 | 1 | | |
| 2.81 V | 2.75 V | | 1 | 1 | 1 | 1 | | |
| 2.92 V | 2.86 V | | 1 | 1 | 1 | 0 | | |
| 3.02 V | 2.96 V | | 1 | 1 | 0 | 1 | | |
| 3.13 V | 3.06 V | | 0 | 1 | 0 | 0 | | |
| – | | Setting of values other than above is prohibited. | | | | | | |

<R> • LVD off (by controlling the externally input reset signal on the $\overline{\text{RESET}}$ pin)

| Detection voltage | | Option byte setting value | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $V_{LVD}$ | | VPOC2 | VPOC1 | VPOC0 | LVIS1 | LVIS0 | Mode setting | |
| Rising edge | Falling edge | | | | | | LVIMDS1 | LVIMDS0 |
| – | – | 1 | × | × | × | × | × | 1 |
| – | | Setting of values other than above is prohibited. | | | | | | |

**Note** Set the same value as 000C1H to 010C1H when the boot swap operation is used because 000C1H is replaced by 010C1H.

**Cautions 1. Be sure to set bit 4 to "1".**

<R> **2. After power is supplied, the reset state must be retained until the operating voltage becomes in the range defined in 29.4 or 30.4 AC Characteristics. This is done by utilizing the voltage detection circuit or controlling the externally input reset signal. After the power supply is turned off, this LSI should be placed in the STOP mode, or placed in the reset state by utilizing the voltage detection circuit or controlling the externally input reset signal, before the voltage falls below the operating range. The range of operating voltage varies with the setting of the user option byte (000C2H or 010C2H).**

**Remarks 1.** ×: don't care

**2.** For details on the LVD circuit, see **CHAPTER 21 VOLTAGE DETECTOR**.

<R> **3.** The detection voltage is a typical value. For details, see **29.6.4** or **30.6.4 LVD circuit characteristics**.

**Figure 24-3. Format of Option Byte (000C2H/010C2H)**

Address: 000C2H/010C2H[Note]

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CMODE1 | CMODE0 | 1 | 0 | FRQSEL3 | FRQSEL2 | FRQSEL1 | FRQSEL0 |

| CMODE1 | CMODE0 | Setting of flash operation mode | | |
|---|---|---|---|---|
| | | | Operating frequency range | Operating voltage range |
| 0 | 0 | LV (low voltage main) mode | 1 to 4 MHz | 1.6 to 3.6 V |
| 1 | 0 | LS (low speed main) mode | 1 to 8 MHz | 1.8 to 3.6 V |
| 1 | 1 | HS (high speed main) mode | 1 to 16 MHz | 2.4 to 3.6 V |
| | | | 1 to 32 MHz | 2.7 to 3.6 V |
| Other than above | | Setting prohibited | | |

| FRQSEL3 | FRQSEL2 | FRQSEL1 | FRQSEL0 | Frequency of the high-speed on-chip oscillator |
|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 32 MHz |
| 0 | 0 | 0 | 0 | 24 MHz |
| 1 | 0 | 0 | 1 | 16 MHz |
| 0 | 0 | 0 | 1 | 12 MHz |
| 1 | 0 | 1 | 0 | 8 MHz |
| 0 | 0 | 1 | 0 | 6 MHz |
| 1 | 0 | 1 | 1 | 4 MHz |
| 0 | 0 | 1 | 1 | 3 MHz |
| 1 | 1 | 0 | 0 | 2 MHz |
| 1 | 1 | 0 | 1 | 1 MHz |
| Other than above | | | | Setting prohibited |

**Note** Set the same value as 000C2H to 010C2H when the boot swap operation is used because 000C2H is replaced by 010C2H.

**Cautions 1. Be sure to set bits 4 and 5 to "10B".**

<R> **2. The ranges of operation frequency and operation voltage vary depending on the flash operation mode. For details, see 29.4 or 30.4 AC Characteristics.**

## 24.3 Format of On-chip Debug Option Byte

The format of on-chip debug option byte is shown below.

**Figure 24-4. Format of On-chip Debug Option Byte (000C3H/010C3H)**

Address: 000C3H/010C3H[Note]

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| OCDENSET | 0 | 0 | 0 | 0 | 1 | 0 | OCDERSD |

| OCDENSET | OCDERSD | Control of on-chip debug operation |
|---|---|---|
| 0 | 0 | Disables on-chip debug operation. |
| 0 | 1 | Setting prohibited |
| 1 | 0 | Enables on-chip debugging.<br>Erases data of flash memory in case of failures in authenticating on-chip debug security ID. |
| 1 | 1 | Enables on-chip debugging.<br>Does not erases data of flash memory in case of failures in authenticating on-chip debug security ID. |

**Note** Set the same value as 000C3H to 010C3H when the boot swap operation is used because 000C3H is replaced by 010C3H.

**Caution Bits 7 and 0 (OCDENSET and OCDERSD) can only be specified a value.**
**Be sure to set 000010B to bits 6 to 1.**

**Remark** The value on bits 3 to 1 will be written over when the on-chip debug function is in use and thus it will become unstable after the setting.
However, be sure to set the default values (0, 1, and 0) to bits 3 to 1 at setting.

## 24.4 Setting of Option Byte

<R> The user option byte and on-chip debug option byte can be set using the link option, in addition to describing to the source. When doing so, the contents set by using the link option take precedence, even if descriptions exist in the source, as mentioned below.

A software description example of the option byte setting is shown below.

```
OPT   CSEG        OPT_BYTE
      DB          36H     ; Does not use interval interrupt of watchdog timer,
                          ; Enables watchdog timer operation,
                          ; Window open period of watchdog timer is 50%,
                          ; Overflow time of watchdog timer is 2^9/fIL,
                          ; Stops watchdog timer operation during HALT/STOP mode
      DB          1AH     ; Select 1.63 V for VLVDL
                          ; Select rising edge 1.77 V, falling edge 1.73 V for VLVDH
                          ; Select the interrupt & reset mode as the LVD operation mode
      DB          2DH     ; Select the LV (low voltage main) mode as the flash operation mode
                            and 1 MHz as the frequency of the high-speed on-chip oscillator
      DB          85H     ; Enables on-chip debug operation, does not erase flash memory
                            data when security ID authorization fails
```

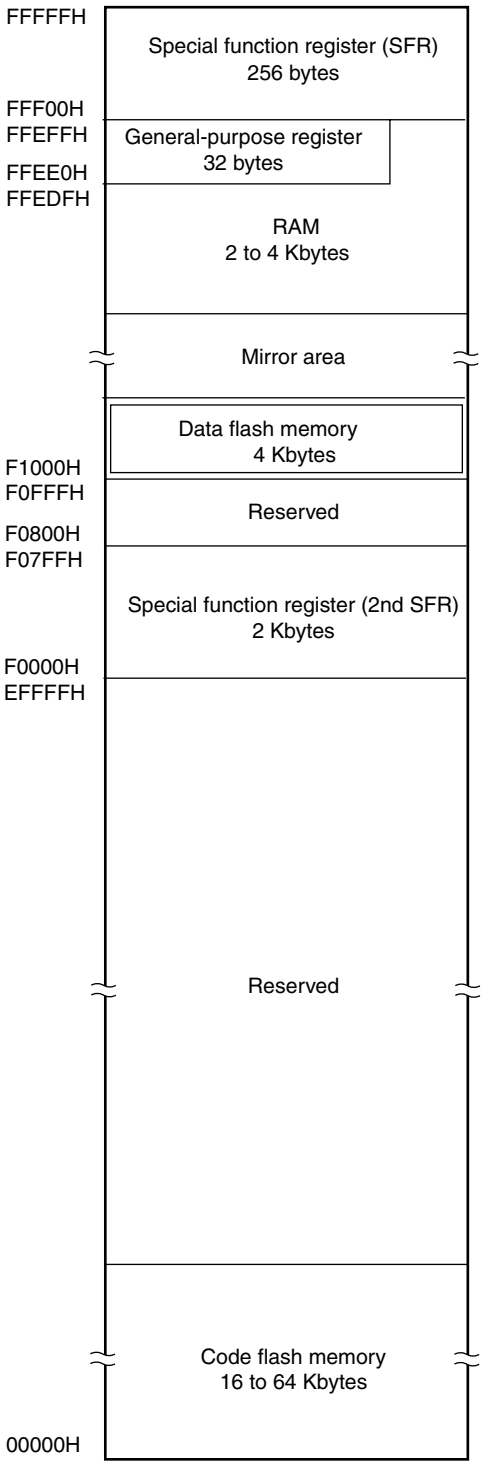
When the boot swap function is used during self programming, 000C0H to 000C3H is switched to 010C0H to 010C3H. Describe to 010C0H to 010C3H, therefore, the same values as 000C0H to 000C3H as follows.

```
OPT2  CSEG   AT   010C0H
      DB          36H     ; Does not use interval interrupt of watchdog timer,
                          ; Enables watchdog timer operation,
                          ; Window open period of watchdog timer is 50%,
                          ; Overflow time of watchdog timer is 2^10/fIL,
                          ; Stops watchdog timer operation during HALT/STOP mode
      DB          1AH     ; Select 1.63 V for VLVDL
                          ; Select rising edge 1.77 V, falling edge 1.73 V for VLVDH
                          ; Select the interrupt & reset mode as the LVD operation mode
      DB          2DH     ; Select the LV (low main voltage) mode as the flash operation mode
                            and 1 MHz as the frequency of the high-speed on-chip oscillator
      DB          85H     ; Enables on-chip debug operation, does not erase flash memory
                            data when security ID authorization fails
```

**Caution To specify the option byte by using assembly language, use OPT_BYTE as the relocation attribute name of the CSEG pseudo instruction. To specify the option byte to 010C0H to 010C3H in order to use the boot swap function, use the relocation attribute AT to specify an absolute address.**

# CHAPTER 25 FLASH MEMORY

The RL78 microcontroller incorporates the flash memory to which a program can be written, erased, and overwritten while mounted on the board. The flash memory includes the “code flash memory”, in which programs can be executed, and the “data flash memory”, an area for storing data.



The following methods for programming the flash memory are available.

<R> The code flash memory can be rewritten to through serial programming using a flash memory programmer or an external device (UART communication), or through self-programming.

- Serial programming using flash memory programmer (see **25.4**)
  Data can be written to the flash memory on-board or off-board by using a dedicated flash memory programmer.

- Serial programming using external device (UART communication) (see **25.2**)
  Data can be written to the flash memory on-board through UART communication with an external device (microcontroller or ASIC).

- Self-programming (see **25.6**)
  The user application can execute self-programming of the code flash memory by using the flash self-programming library.

The data flash memory can be rewritten to by using the data flash library during user program execution (background operation). For access and writing to the data flash memory, see **25.8 Data Flash**.

## 25.1 Serial Programming Using Flash Memory Programmer

The following dedicated flash memory programmer can be used to write data to the internal flash memory of the RL78 microcontroller.

- PG-FP5, FL-PR5
- E1 on-chip debugging emulator

Data can be written to the flash memory on-board or off-board, by using a dedicated flash memory programmer.

**(1) On-board programming**

The contents of the flash memory can be rewritten after the RL78 microcontroller has been mounted on the target system. The connectors that connect the dedicated flash memory programmer must be mounted on the target system.

**(2) Off-board programming**

Data can be written to the flash memory with a dedicated program adapter (FA series) before the RL78 microcontroller is mounted on the target system.

**Remark** FL-PR5 and FA series are products of Naito Densei Machida Mfg. Co., Ltd.

<R> **Table 25-1. Wiring between RL78/G1A and Dedicated Flash Memory Programmer**

| Pin Configuration of Dedicated Flash Memory Programmer | | | | Pin Name | Pin No. | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | 25-pin | 32-pin | 48-pin | 64-pin | |
| Signal Name | | I/O | Pin Function | | FLGA (3x3) | WQFN (5x5) | LQFP (7x7), WQFN (7x7) | LQFP (10x10) | FBGA (4x4) |
| PG-FP5, FL-PR5 | E1 on-chip debugging emulator | | | | | | | | |
| – | TOOL0 | I/O | Transmit/ receive signal | TOOL0/ P40 | A5 | 1 | 39 | 5 | D6 |
| SI/RxD | – | I/O | Transmit/ receive signal | | | | | | |
| – | $\overline{\text{RESET}}$ | Output | Reset signal | $\overline{\text{RESET}}$ | B5 | 2 | 40 | 6 | E7 |
| /RESET | – | Output | | | | | | | |
| $V_{DD}$ | | I/O | $V_{DD}$ voltage generation/ power monitoring | $V_{DD}$ | B3 | 8 | 48 | 15 | B7 |
| GND | | – | Ground | $V_{SS}$ | B2 | 7 | 47 | 13 | C7 |
| | | | | $EV_{SS}$ | – | – | – | 14 | B8 |
| | | | | REGC[Note] | A2 | 6 | 46 | 12 | D7 |
| $EMV_{DD}$ | | – | Driving power for TOOL0 pin | $V_{DD}$ | B3 | 8 | 48 | – | – |
| | | | | $EV_{DD}$ | – | – | – | 16 | A8 |

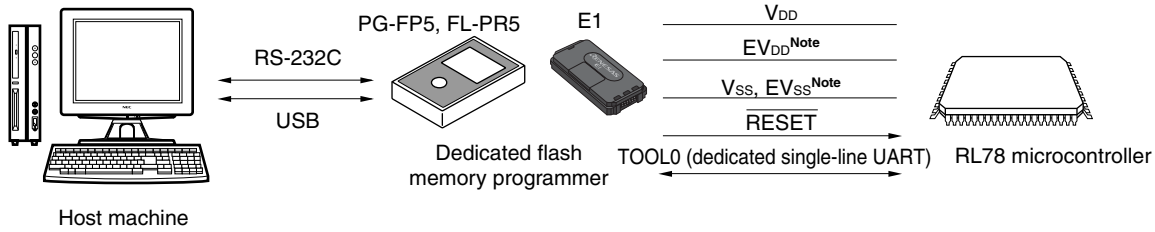
**Note** Connect REGC pin to ground via a capacitor (0.47 to 1 $\mu$F).

**Remark** Pins that are not indicated in the above table can be left open when using the flash memory programmer for flash programming.

### 25.1.1 Programming environment

The environment required for writing a program to the flash memory of the RL78 microcontroller is illustrated below.

<R> **Figure 25-1. Environment for Writing Program to Flash Memory**




**Note 64-pin product only.**

A host machine that controls the dedicated flash memory programmer is necessary.

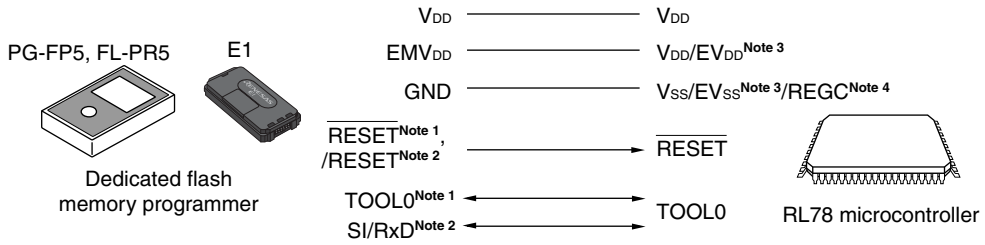
To interface between the dedicated flash memory programmer and the RL78 microcontroller, the TOOL0 pin is used for manipulation such as writing and erasing via a dedicated single-line UART.

### 25.1.2 Communication mode

Communication between the dedicated flash memory programmer and the RL78 microcontroller is established by serial communication using the TOOL0 pin via a dedicated single-line UART of the RL78 microcontroller.

Transfer rate: 1 M, 500 k, 250 k, 115.2 kbps

<R> **Figure 25-2. Communication with Dedicated Flash Memory Programmer**




**Notes 1.** When using E1 on-chip debugging emulator.

**2.** When using PG-FP5 or FL-PR5.

**3.** 64- pin product only.

**4.** Connect REGC pin to ground via a capacitor (0.47 to 1 $\mu$F).

The dedicated flash memory programmer generates the following signals for the RL78 microcontroller. See each manual of PG-FP5, FL-PR5, or E1 on-chip debugging emulator for details.

<R> **Table 25-2. Pin Connection**

| Dedicated Flash Memory Programmer | | | | RL78 Microcontroller |
|---|---|---|---|---|
| Signal Name | | I/O | Pin Function | Pin Name |
| PG-FP5, FL-PR5 | E1 On-chip Debugging Emulator | | | |
| $V_{DD}$ | | I/O | $V_{DD}$ voltage generation/power monitoring | $V_{DD}$ |
| GND | | – | Ground | $V_{SS}$, $EV_{SS0}$, REGC[Note 1] |
| $EMV_{DD}$ | | – | Driving power for TOOL0 pin | $V_{DD}$ or $EV_{DD0}$[Note 2] |
| /RESET | – | Output | Reset signal | $\overline{RESET}$ |
| – | $\overline{RESET}$ | Output | | |
| – | TOOL0 | I/O | Transmit/receive signal | TOOL0 |
| SI/RxD | – | I/O | Transmit/receive signal | |

**Notes 1.** Connect REGC pin to ground via a capacitor (0.47 to 1 μF).

**2.** Pins to be connected differ with the product. For details, see **Table 25-1**.

## 25.2 Serial Programming Using External Device (that Incorporates UART)

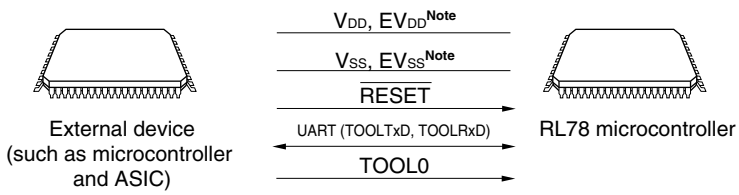
On-board data writing to the internal flash memory is possible by using the RL78 microcontroller and an external device (a microcontroller or ASIC) connected to a UART.

On the development of flash memory programmer by user, refer to the **RL78 Microcontrollers (RL78 Protocol A) Programmer Edition Application Note (R01AN0815)**.

### 25.2.1 Programming environment

The environment required for writing a program to the flash memory of the RL78 microcontroller is illustrated below.

<R> **Figure 25-3. Environment for Writing Program to Flash Memory**




**Note** 64-pin product only.

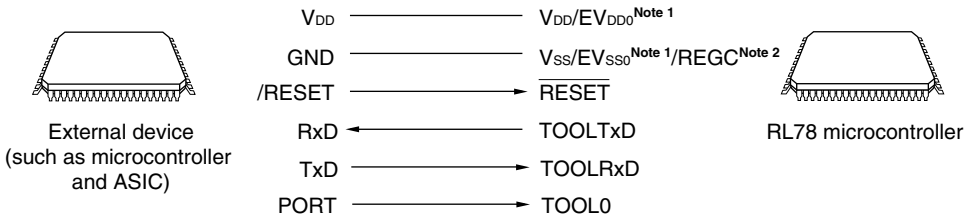
Processing to write data to or erase data from the RL78 microcontroller by using an external device is performed on-board. Off-board writing is not possible.

### 25.2.2 Communication mode

Communication between the external device and the RL78 microcontroller is established by serial communication using the TOOLTxD and TOOLRxD pins via the dedicated UART of the RL78 microcontroller.

Transfer rate: 1 M, 500 k, 250 k, 115.2kbps

<R> **Figure 25-4. Communication with External Device**




**Notes 1.** 64-pin product only.

**2.** Connect REGC pin to ground via a capacitor (0.47 to 1 μF).

The external device generates the following signals for the RL78 microcontroller.

<R> **Table 25-3. Pin Connection**

| External Device | | | RL78 Microcontroller |
|---|---|---|---|
| Signal Name | I/O | Pin Function | Pin Name |
| $V_{DD}$ | I/O | $V_{DD}$ voltage generation/power monitoring | $V_{DD}$, $EV_{DD0}$ |
| GND | – | Ground | $V_{SS}$, $EV_{SS0}$, REGC[Note] |
| RESETOUT | Output | Reset signal output | $\overline{RESET}$ |
| RxD | Input | Receive signal | TOOLTxD |
| TxD | Output | Transmit signal | TOOLRxD |
| PORT | Output | Mode signal | TOOL0 |

**Note** Connect REGC pin to ground via a capacitor (0.47 to 1 μF).

## 25.3 Connection of Pins on Board

To write the flash memory on-board by using the flash memory programmer, connectors that connect the dedicated flash memory programmer must be provided on the target system. First provide a function that selects the normal operation mode or flash memory programming mode on the board.

When the flash memory programming mode is set, all the pins not used for programming the flash memory are in the same status as immediately after reset. Therefore, if the external device does not recognize the state immediately after reset, the pins must be handled as described below.

<R> **Remark** For the flash memory programming mode, see **25.4.2 Flash memory programming mode**.

### 25.3.1 P40/TOOL0 pin

In the flash memory programming mode, connect this pin to the dedicated flash memory programmer via an external 1 kΩ pull-up resistor.

When this pin is used as the port pin, use that by the following method.

<R> When used as an input pin: Input of low-level is prohibited for $t_{HD}$ period after external pin reset release. However, when this pin is used via pull-down resistors, use the 500 kΩ or more resistors.

When used as an output pin: When this pin is used via pull-down resistors, use the 500 kΩ or more resistors.

<R> **Remarks 1.** $t_{HD}$: How long to keep the TOOL0 pin at the low level from when the external and internal resets end for setting of the flash memory programming mode (see **29.9** or **30.9 Timing Specs for Switching Flash Memory Programming Modes**)

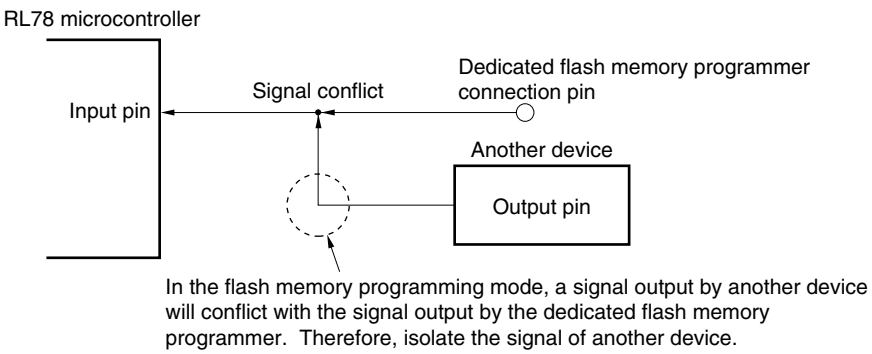
**2.** The SAU and IICA pins are not used for communication between the RL78 microcontroller and dedicated flash memory programmer, because single-line UART (TOOL0 pin) is used.

### 25.3.2 $\overline{\text{RESET}}$ pin

Signal conflict will occur if the reset signal of the dedicated flash memory programmer and external device are connected to the $\overline{\text{RESET}}$ pin that is connected to the reset signal generator on the board. To prevent this conflict, isolate the connection with the reset signal generator.

The flash memory will not be correctly programmed if the reset signal is input from the user system while the flash memory programming mode is set. Do not input any signal other than the reset signal of the dedicated flash memory programmer and external device.

**Figure 25-5. Signal Conflict ($\overline{\text{RESET}}$ Pin)**



### 25.3.3 Port pins

When the flash memory programming mode is set, all the pins not used for flash memory programming enter the same status as that immediately after reset. If external devices connected to the ports do not recognize the port status immediately after reset, the port pin must be connected to either $V_{DD}$ or $EV_{DD0}$, or $V_{SS}$ or $EV_{SS0}$, via a resistor.

### 25.3.4 REGC pin

<R> Connect the REGC pin to GND via a capacitor having excellent characteristics (0.47 to 1 $\mu$F) in the same manner as during normal operation. Also, use a capacitor with good characteristics, since it is used to stabilize internal voltage.

### 25.3.5 X1 and X2 pins

Connect X1 and X2 in the same status as in the normal operation mode.

**Remark** In the flash memory programming mode, the high-speed on-chip oscillator clock ($f_{IH}$) is used.

### 25.3.6 Power supply

To use the supply voltage output of the flash memory programmer, connect the $V_{DD}$ pin to $V_{DD}$ of the flash memory programmer, and the $V_{SS}$ pin to GND of the flash memory programmer.

To use the on-board supply voltage, connect in compliance with the normal operation mode.

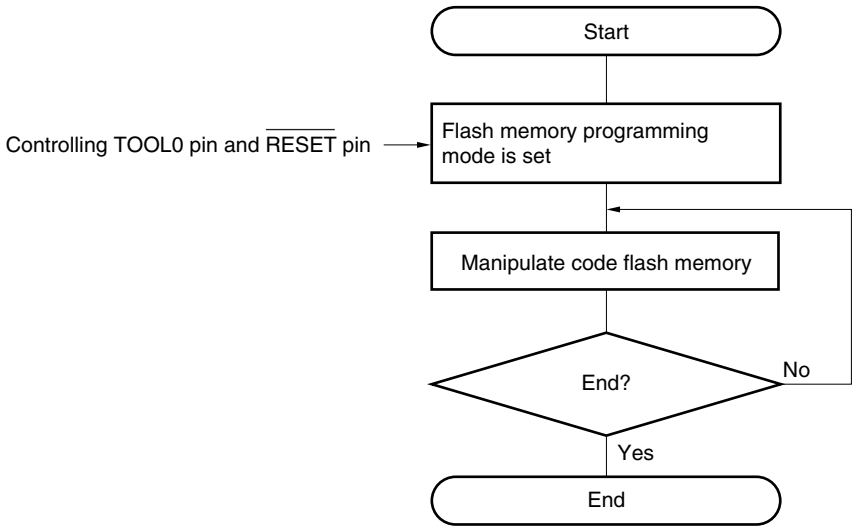
However, when writing to the flash memory by using the flash memory programmer and using the on-board supply voltage, be sure to connect the $V_{DD}$ and $V_{SS}$ pins to $V_{DD}$ and GND of the flash memory programmer to use the power monitor function with the flash memory programmer.

## 25.4 Serial Programming Method

### 25.4.1 Serial programming procedure

The following figure illustrates a flow for rewriting the code flash memory through serial programming.

**Figure 25-6. Code Flash Memory Manipulation Procedure**



### 25.4.2 Flash memory programming mode

To rewrite the contents of the code flash memory through serial programming, specify the flash memory programming mode. To enter the mode, set as follows.

<Serial programming using the dedicated flash memory programmer>

Connect the RL78 microcontroller to a dedicated flash memory programmer. Communication from the dedicated flash memory programmer is performed to automatically switch to the flash memory programming mode.

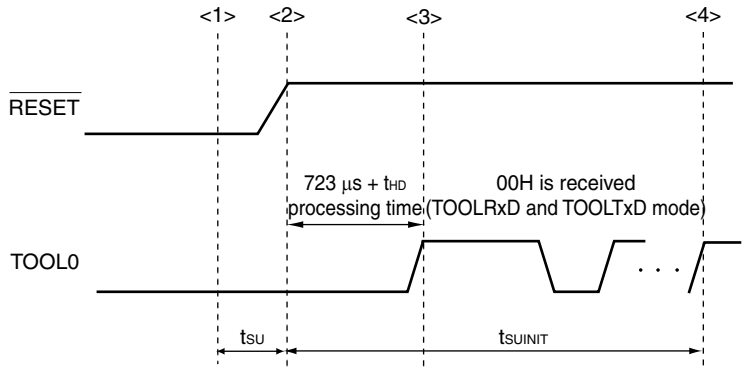
<R> <Serial programming using an external device (UART communication)>

Set the TOOL0 pin to the low level, and then cancel the reset (see **Table 25-4**). After that, enter flash memory programming mode according to the procedures <1> to <4> shown in **Figure 25-7**. For details, refer to the **RL78 Microcontrollers (RL78 Protocol A) Programmer Edition Application Note (R01AN0815)**.

**Table 25-4. Relationship Between TOOL0 Pin and Operation Mode After Reset Release**

| TOOL0 | Operation Mode |
|---|---|
| $EV_{DD0}$ | Normal operation mode |
| 0 V | Flash memory programming mode |

<R> **Figure 25-7. Setting of Flash Memory Programming Mode**




<1> The low level is input to the TOOL0 pin.
<2> The external reset ends (POR and LVD reset must end before the external reset ends.).
<3> The TOOL0 pin is set to the high level.
<4> Baud rate setting by UART reception is completed.

**Remark** $t_{SUINIT}$: The segment shows that it is necessary to finish specifying the initial communication settings within 100 ms from when the resets end.
$t_{SU}$: How long from when the TOOL0 pin is placed at the low level until an external reset ends
$t_{HD}$: How long to keep the TOOL0 pin at the low level from when the external and internal resets end (the flash firmware processing time is excluded)

For details, see **29.9** or **30.9 Timing Specs for Switching Flash Memory Programming Modes**.

There are two flash memory programming modes: wide voltage mode and full speed mode. The supply voltage value applied to the microcontroller during write operations and the setting information of the user option byte for setting of the flash memory programming mode determine which mode is selected.

When a dedicated flash memory programmer is used for serial programming, setting the voltage on GUI selects the mode automatically.

<R> **Table 25-5. Programming Modes and Voltages at Which Data Can Be Written, Erased, or Verified**

| Power Supply Voltage ($V_{DD}$) | User Option Byte Setting for Switching to Flash Memory Programming Mode | | Flash Programming Mode |
|---|---|---|---|
| | Flash Operation Mode | Operating Frequency | |
| 2.7 V ≤ $V_{DD}$ ≤ 3.6 V | Blank state | | Full speed mode |
| | HS (high speed main) mode | 1 MHz to 32 MHz | Full speed mode |
| | LS (low speed main) mode | 1 MHz to 8 MHz | Wide voltage mode |
| | LV (low voltage main) mode | 1 MHz to 4 MHz | Wide voltage mode |
| 2.4 V ≤ $V_{DD}$ < 2.7 V | Blank state | | Full speed mode |
| | HS (high speed main) mode | 1 MHz to 16 MHz | Full speed mode |
| | LS (low speed main) mode | 1 MHz to 8 MHz | Wide voltage mode |
| | LV (low voltage main) mode | 1 MHz to 4 MHz | Wide voltage mode |
| 1.8 V ≤ $V_{DD}$ < 2.4 V | Blank state | | Wide voltage mode |
| | LS (low speed main) mode | 1 MHz to 8 MHz | Wide voltage mode |
| | LV (low voltage main) mode | 1 MHz to 4 MHz | Wide voltage mode |

**Remarks 1.** Using both the wide voltage mode and full speed mode imposes no restrictions on writing, erasing, or verification.

**2.** For details about communication commands, see **25.4.4 Communication commands**.

### 25.4.3 Selecting communication mode

Communication modes of the RL78 microcontroller are as follows.

**Table 25-6. Communication Modes**

| Communication Mode | Standard Setting[Note 1] | | | | Pins Used |
|---|---|---|---|---|---|
| | Port | Speed[Note 2] | Frequency | Multiply Rate | |
| 1-line UART (when flash memory programmer is used, or when external device is used) | UART | 115200 bps, 250000 bps, 500000 bps, 1 Mbps | – | – | TOOL0 |
| Dedicated UART (when external device is used) | UART | 115200 bps, 250000 bps, 500000 bps, 1 Mbps | – | – | TOOLTxD, TOOLRxD |

**Notes 1.** Selection items for Standard settings on GUI of the flash memory programmer.

**2.** Because factors other than the baud rate error, such as the signal waveform slew, also affect UART communication, thoroughly evaluate the slew as well as the baud rate error.

### <R> 25.4.4 Communication commands

The RL78 microcontroller executes serial programming through the commands listed in **Table 25-7.**

The signals sent from the dedicated flash memory programmer or external device to the RL78 microcontroller are called commands, and programming functions corresponding to the commands are executed. For details, refer to the **RL78 Microcontrollers (RL78 Protocol A) Programmer Edition Application Note (R01AN0815)**.

**Table 25-7. Flash Memory Control Commands**

<R>

| Classification | Command Name | Function |
|---|---|---|
| Verify | Verify | Compares the contents of a specified area of the flash memory with data transmitted from the programmer. |
| Erase | Block Erase | Erases a specified area in the flash memory. |
| Blank check | Block Blank Check | Checks if a specified block in the flash memory has been correctly erased. |
| Write | Programming | Writes data to a specified area in the flash memory. |
| Getting information | Silicon Signature | Gets the RL78 microcontroller information (such as the part number, flash memory configuration, and programming firmware version). |
| | Checksum | Gets the checksum data for a specified area. |
| Security | Security Set | Sets security information. |
| | Security Get | Gets security information. |
| | Security Release | Release setting of prohibition of writing. |
| Others | Reset | Used to detect synchronization status of communication. |
| | Baud Rate Set | Sets baud rate when UART communication mode is selected. |

**Note** Confirm that no data has been written to the write area. Because data cannot be erased after block erase is prohibited, do not write data if the data has not been erased.

Product information (such as product name and firmware version) can be obtained by executing the “Silicon Signature” command.

**Table 25-8** is a list of signature data and **Table 25-9** shows an example of signature data.

**Table 25-8. Signature Data List**

| Field Name | Description | Number of Transmit Data |
|---|---|---|
| Device code | The serial number assigned to the device | 3 bytes |
| Device name | Device name (ASCII code) | 10 bytes |
| Code flash memory area last address | Last address of code flash memory area<br>(Sent from lower address.<br>Example: 00000H to 0FFFFH (64 KB) → FFH, 1FH, 00H) | 3 bytes |
| Data flash memory area last address | Last address of data flash memory area<br>(Sent from lower address.<br>Example: F1000H to F1FFFH (4 KB) → FFH, 1FH, 0FH) | 3 bytes |
| Firmware version | Version information of firmware for programming<br>(Sent from upper address.<br>Example: From Ver. 1.23 → 01H, 02H, 03H) | 3 bytes |

**Table 25-9. Example of Signature Data**

| Field Name | Description | Number of Transmit Data | Data (Hexadecimal) |
|---|---|---|---|
| Device code | RL78 protocol A | 3 bytes | 10<br>00<br>06 |
| Device name | R5F10ELE | 10 bytes | 52 = “R”<br>35 = “5”<br>46 = “F”<br>31 = “1”<br>30 = “0”<br>45 = “E”<br>4C = “L”<br>45 = “E”<br>20 = “ ”<br>20 = “ ” |
| Code flash memory area last address | Code flash memory area<br>00000H to 0FFFFH (64 KB) | 3 bytes | FF<br>FF<br>00 |
| Data flash memory area last address | Data flash memory area<br>F1000H to F1FFFH (4 KB) | 3 bytes | FF<br>1F<br>0F |
| Firmware version | Ver.1.23 | 3 bytes | 01<br>02<br>03 |

## <R> 25.5 Processing Time for Each Command When PG-FP5 Is in Use (Reference Value)

The following shows the processing time for each command (reference value) when PG-FP5 is used as a dedicated flash memory programmer.

**Table 25-10. Processing Time for Each Command When PG-FP5 Is in Use (Reference Value)**

| PG-FP5 Command | Code Flash | | | |
|---|---|---|---|---|
| | 16 KB | 32 KB | 48 KB | 64 KB |
| Erasing | 1 s | 1 s | 1 s | 1.5 s |
| Writing | 1.5 s | 1.5 s | 2 s | 2.5 s |
| Verification | 1.5 s | 1.5 s | 2 s | 2 s |
| Writing after erasing | 1.5 s | 2 s | 2.5 s | 3 s |

**Remark** The command processing times (reference values) shown in the table are typical values under the following conditions.
Port: TOOL0 (single-line UART)
Speed: 1,000,000 bps
Mode: Full speed mode (flash operation mode: HS (high speed main) mode)

## <R> 25.6 Self-Programming

The RL78 microcontroller supports a self-programming function that can be used to rewrite the code flash memory via a user program. Because this function allows a user application to rewrite the code flash memory by using the flash self-programming library, it can be used to upgrade the program in the field.

**Cautions 1. The self-programming function cannot be used when the CPU operates with the subsystem clock.**

**2. To prohibit an interrupt during self-programming, in the same way as in the normal operation mode, execute the flash self-programming library in the state where the IE flag is cleared (0) by the DI instruction. To enable an interrupt, clear (0) the interrupt mask flag to accept in the state where the IE flag is set (1) by the EI instruction, and then execute the flash self-programming library.**

**3. The high-speed on-chip oscillator should be kept operating during self-programming. If it is kept stopping, the high-speed on-chip oscillator clock should be operated (HIOSTOP = 0). The flash self-programming library should be executed after 30 μs have elapsed.**

**Remarks 1.** For details of the self-programming function, refer to the **RL78 Microcontroller Flash Self Programming Library Type01 User's Manual (R01AN0350)**.

**2.** For details of the time required to execute self-programming, see the notes on use that accompany the flash self-programming library tool.

The self-programming function has two flash memory programming modes; wide voltage mode and full speed mode.

Specify the mode that corresponds to the flash operation mode specified in bits CMODE1 and CMODE0 in option byte 000C2H.

Specify the full speed mode when the HS (high speed main) mode is specified. Specify the wide voltage mode when the LS (low speed main) mode or LV (low voltage main) mode is specified.

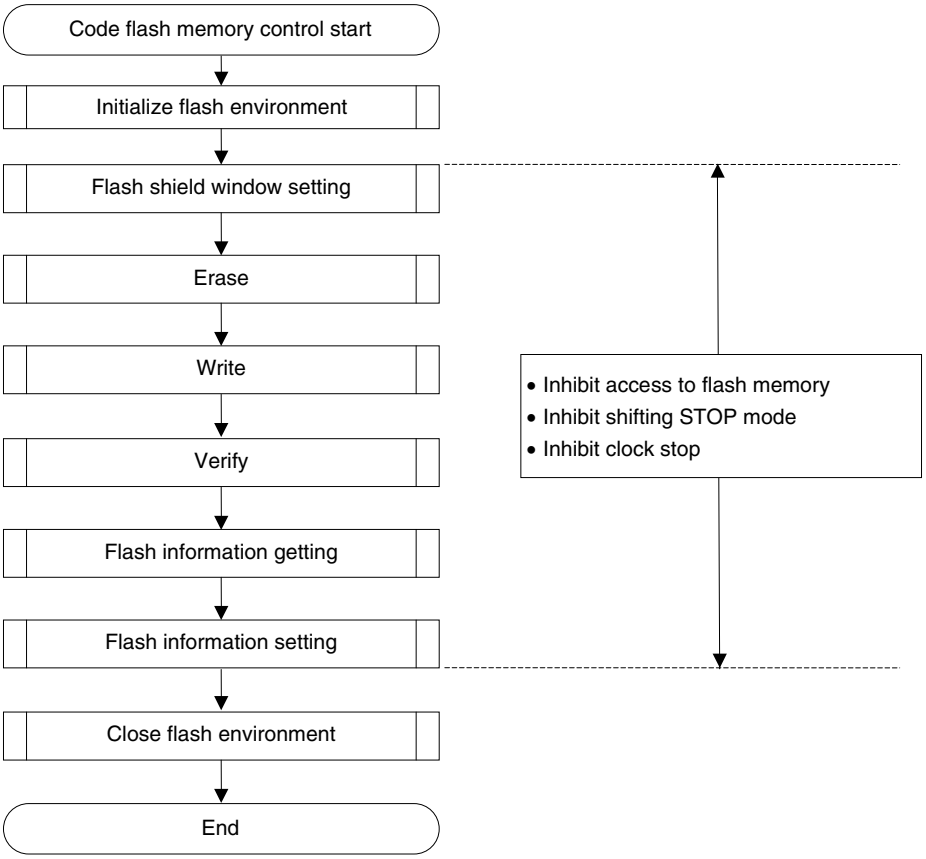
If the argument fsl_flash_voltage_u08 is 00H when the FSL_Init function of the flash self-programming library provided by Renesas Electronics is executed, full speed mode is specified. If the argument is other than 00H, the wide voltage mode is specified.

**Remark** Using both the wide voltage mode and full speed mode imposes no restrictions on writing, erasing, or verification.

### 25.6.1 Self-programming procedure

The following figure illustrates a flow for rewriting the code flash memory by using a flash self-programming library.

**Figure 25-8. Flow of Self-Programming (Rewriting Flash Memory)**



### 25.6.2 Boot swap function

If rewriting the boot area failed by temporary power failure or other reasons, restarting a program by resetting or overwriting is disabled due to data destruction in the boot area.

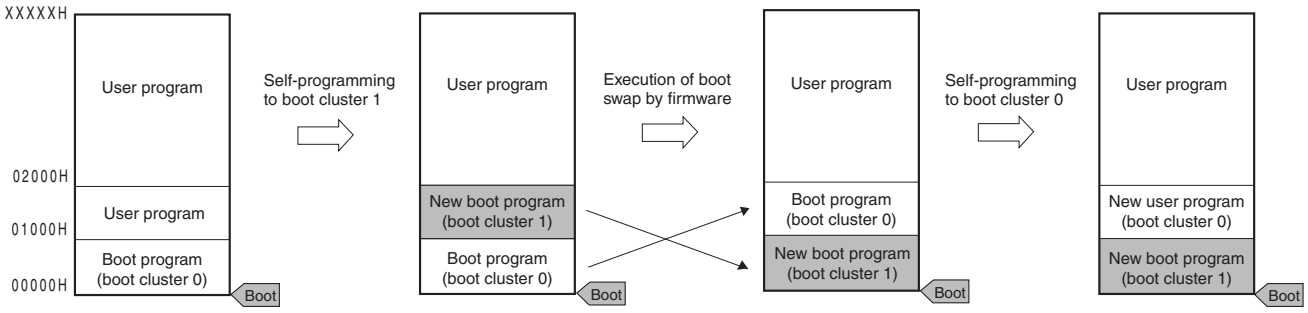
The boot swap function is used to avoid this problem.

Before erasing boot cluster 0[Note], which is a boot area, by self-programming, write a new boot program to boot cluster 1 in advance. When the program has been correctly written to boot cluster 1, swap this boot cluster 1 and boot cluster 0 by using the set information function of the firmware of the RL78 microcontroller, so that boot cluster 1 is used as a boot area. After that, erase or write the original area, boot cluster 0.

As a result, even if a power failure occurs while the area is being rewritten, the program is executed correctly because it is booted from boot cluster 1 to be swapped when the program is reset and started next.

**Note** A boot cluster is a 4 KB area and boot clusters 0 and 1 are swapped by the boot swap function.

**Figure 25-9. Boot Swap Function**

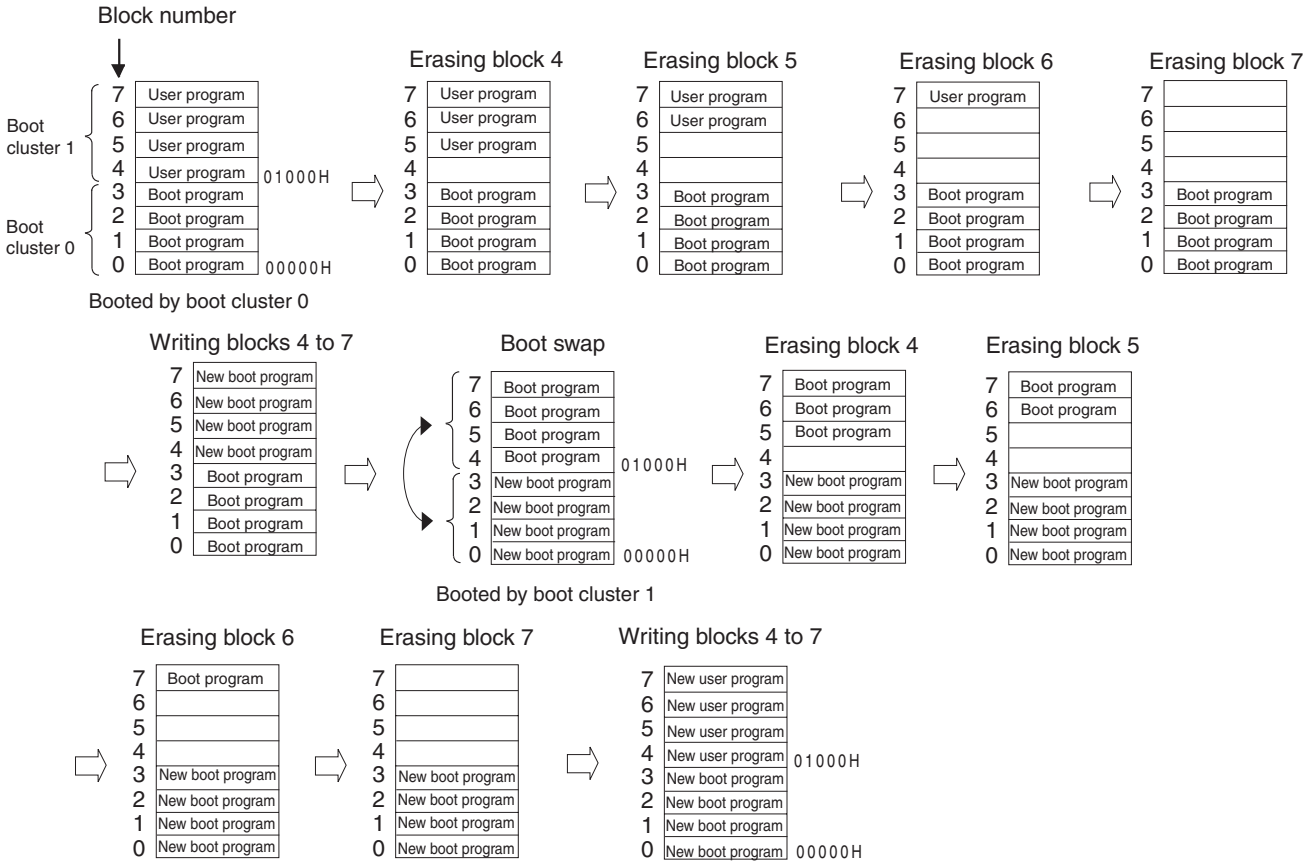



In an example of above figure, it is as follows.

Boot cluster 0: Boot area before boot swap

Boot cluster 1: Boot area after boot swap

**Figure 25-10. Example of Executing Boot Swapping**



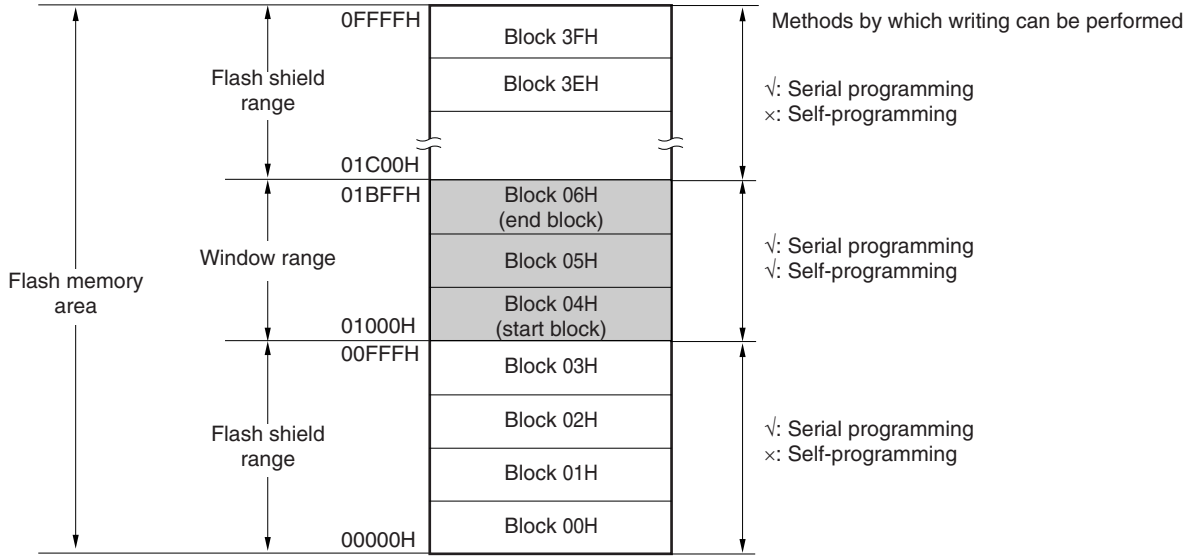
### 25.6.3 Flash shield window function

The flash shield window function is provided as one of the security functions for self-programming. It disables writing to and erasing areas outside the range specified as a window only during self-programming.

The window range can be set by specifying the start and end blocks. The window range can be set or changed during both serial programming and self-programming.

Writing to and erasing areas outside the window range are disabled during self-programming. During serial programming, however, areas outside the range specified as a window can be written and erased.

**Figure 25-11. Flash Shield Window Setting Example**
**(Target Devices: R5F10ELE, Start Block: 04H, End Block: 06H)**




**Cautions 1. If the rewrite-prohibited area of the boot cluster 0 overlaps with the flash shield window range, prohibition to rewrite the boot cluster 0 takes priority.**

**2. The flash shield window can only be used for the code flash memory (and is not supported for the data flash memory).**

**Table 25-11. Relationship Between Flash Shield Window Function Setting/Change Methods and Commands**

| Programming Conditions | Window Range Setting/Change Methods | Execution Commands | |
|---|---|---|---|
| | | Block Erase | Write |
| Self-programming | Specify the starting and ending blocks by the flash self-programming library. | Block erasing is enabled only within the window range. | Writing is enabled only within the range of window range. |
| Serial programming | Specify the starting and ending blocks on GUI of dedicated flash memory programmer, etc. | Block erasing is enabled also outside the window range. | Writing is enabled also outside the window range. |

**Remark** See **25.7 Security Settings** to prohibit writing/erasing during serial programming.

## 25.7 Security Settings

The RL78 microcontroller supports a security function that prohibits rewriting the user program written to the internal flash memory, so that the program cannot be changed by an unauthorized person.

The operations shown below can be performed using the Security Set command.

- Disabling block erase
  Execution of the block erase command for a specific block in the flash memory is prohibited during serial programming. However, blocks can be erased by means of self-programming.

- Disabling write
  Execution of the write command for entire blocks in the code flash memory is prohibited during serial programming. However, blocks can be written by means of self-programming.
<R> After the setting of prohibition of writing is specified, releasing the setting by the Security Release command is enabled by a reset.

- Disabling rewriting boot cluster 0
  Execution of the block erase command and write command on boot cluster 0 (00000H to 00FFFH) in the code flash memory is prohibited by this setting.

The block erase, write commands and rewriting boot cluster 0 are enabled by the default setting when the flash memory is shipped. Security can be set by serial programming and self-programming. Each security setting can be used in combination.

**Table 25-12** shows the relationship between the erase and write commands when the RL78 microcontroller security function is enabled.

**Caution The security function of the dedicated flash programmer does not support self-programming.**

**Remark** To prohibit writing and erasing during self-programming, use the flash shield window function (see **25.6.3** for details).

**Table 25-12. Relationship Between Enabling Security Function and Command**

**(1) During serial programming**

| Valid Security | Executed Command | |
|---|---|---|
| | Block Erase | Write |
| Prohibition of block erase | Blocks cannot be erased. | Can be performed.[Note] |
| Prohibition of writing | Blocks can be erased. | Cannot be performed. |
| Prohibition of rewriting boot cluster 0 | Boot cluster 0 cannot be erased. | Boot cluster 0 cannot be written. |

**Note** Confirm that no data has been written to the write area. Because data cannot be erased after block erase is prohibited, do not write data if the data has not been erased.

**(2) During self-programming**

| Valid Security | Executed Command | |
|---|---|---|
| | Block Erase | Write |
| Prohibition of block erase | Blocks can be erased. | Can be performed. |
| Prohibition of writing | | |
| Prohibition of rewriting boot cluster 0 | Boot cluster 0 cannot be erased. | Boot cluster 0 cannot be written. |

**Remark** To prohibit writing and erasing during self-programming, use the flash shield window function (see **25.6.3** for detail).

**Table 25-13. Setting Security in Each Programming Mode**

**(1) During serial programming**

| Security | Security Setting | How to Disable Security Setting |
|---|---|---|
| Prohibition of block erase | Set via GUI of dedicated flash memory programmer, etc. | Cannot be disabled after set. |
| Prohibition of writing | | Set via GUI of dedicated flash memory programmer, etc. |
| Prohibition of rewriting boot cluster 0 | | Cannot be disabled after set. |

<R> **Caution Releasing the setting of prohibition of writing is enabled only when the security is not set as the block erase prohibition and the boot cluster 0 rewrite prohibition with code flash memory area and data flash memory area being blanks.**

**(2) During self-programming**

| Security | Security Setting | How to Disable Security Setting |
|---|---|---|
| Prohibition of block erase | Set by using flash self-programming library. | Cannot be disabled after set. |
| Prohibition of writing | | Cannot be disabled during self-programming (set via GUI of dedicated flash memory programmer, etc. during serial programming). |
| Prohibition of rewriting boot cluster 0 | | Cannot be disabled after set. |

## 25.8 Data Flash

### 25.8.1 Data flash overview

An overview of the data flash memory is provided below.

- The user program can rewrite the data flash memory by using the data flash library. For details, refer to the **RL78 Family Data Flash Library User's Manual**.
- The data flash memory can also be rewritten to through serial programming using the dedicated flash memory programmer or an external device.
- The data flash can be erased in 1-block (1 KB) units.
- The data flash can be accessed only in 8-bit units.
- The data flash can be directly read by CPU instructions.
- Instructions can be executed from the code flash memory while rewriting the data flash memory (that is, background operation (BGO) is supported).
- Because the data flash memory is an area exclusively used for data, it cannot be used to execute instructions.
- Accessing the data flash memory is not possible while rewriting the code flash memory (during self-programming).
- Manipulating the DFLCTL register is not possible while rewriting the data flash memory.
- Transition to the STOP mode is not possible while rewriting the data flash memory.

<R> **Cautions 1. The data flash memory is stopped after a reset is canceled. The data flash control register (DFLCTL) must be set up in order to use the data flash memory.**

**2. The high-speed on-chip oscillator should be kept operating during data flash rewrite. If it is kept stopping, the high-speed on-chip oscillator clock should be operated (HIOSTOP = 0). The data flash library should be executed after 30 μs have elapsed.**

**Remark** For rewriting the code flash memory via a user program, see **25.6 Self-Programming**.

### 25.8.2 Register controlling data flash memory

#### 25.8.2.1 Data flash control register (DFLCTL)

This register is used to enable or disable accessing to the data flash.

The DFLCTL register is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets this register to 00H.

**Figure 25-12. Format of Data Flash Control Register (DFLCTL)**

Address: F0090H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| DFLCTL | 0 | 0 | 0 | 0 | 0 | 0 | 0 | DFLEN |

| DFLEN | Data flash access control |
|---|---|
| 0 | Disables data flash access |
| 1 | Enables data flash access |

**Caution Manipulating the DFLCTL register is not possible while rewriting the data flash memory.**

### 25.8.3 Procedure for accessing data flash memory

The data flash memory is stopped after a reset ends. To access the data flash, make initial settings according to the following procedure.

<1> Set bit 0 (DFLEN) of the data flash control register (DFLCTL) to 1.
<2> Wait for the setup to finish for software timer, etc.
The time setup takes differs for each flash operation mode for the main clock.
<Setup time for each flash operation mode>
- HS (High speed main): 5 μs
- LS (Low speed main): 720 ns
- LV (Low voltage main): 10 μs

<3> After the wait, the data flash memory can be accessed.

**Cautions 1. Accessing the data flash memory is not possible during the setup time.**

<R> **2. Transition to the STOP mode is not possible during the setup time. To enter the STOP mode during the setup time, clear DFLEN to 0 and then execute the STOP instruction.**

<R> **3. The high-speed on-chip oscillator should be kept operating during data flash rewrite. If it is kept stopping, operate the high-speed on-chip oscillator clock (HIOSTOP = 0) and execute the data flash library after 30 μs have elapsed.**

After initialized, the data flash memory can be read by using a CPU instruction or can be read/written by using a data flash library.

<R> If the DMA controller operates when the data flash memory is accessed, however, follow one of these procedures:

(A) Suspending/forcibly terminating DMA transfer
Before reading the data flash memory, suspend DMA transfer of all the channels used.
After setting the DWAITn bit to 1, however, wait at least for the duration of three clocks ($f_{CLK}$) before reading the data flash memory. After reading the data flash memory, lift the suspension of transfer by clearing the DWAITn bit to 0.
Or, forcibly terminate DMA transfer in accordance with the procedure in 15.5.5 Forced termination by software before reading the data flash memory. Resume DMA transfer after the data flash memory has been read.

(B) Access the data flash memory by using the newest data flash library.

(C) Insertion of NOP
Insert an NOP instruction immediately before the instruction that reads the data flash memory.
<Example>

```
MOVW    HL,!addr16      ; Reads RAM.
NOP                     ; Insert NOP instruction before reading data flash memory.
MOV     A,[DE]          ; Read data flash memory.
```

If a high-level language such as C is used, however, the compiler may generate two instructions for one code. In this case, the NOP instruction is not inserted immediately before the data flash memory read instruction. Therefore, read the data flash memory by (A) or (B) above.

**Remarks 1.** n: DMA channel number (n = 0, 1)
**2.** $f_{CLK}$: CPU/peripheral hardware clock frequency

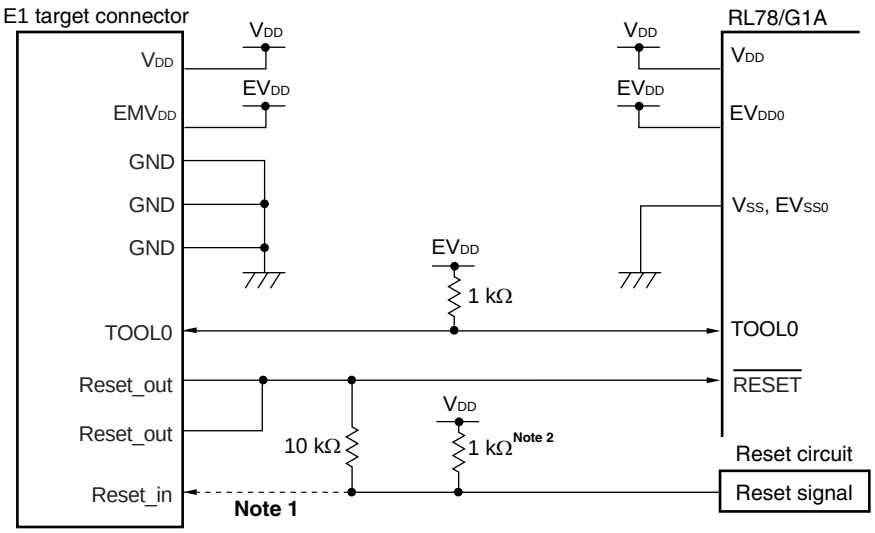
# CHAPTER 26 ON-CHIP DEBUG FUNCTION

## 26.1 Connecting E1 On-Chip Debugging Emulator

The RL78 microcontroller uses the $V_{DD}$, $\overline{RESET}$, TOOL0, and $V_{SS}$ pins to communicate with the host machine via an E1 on-chip debugging emulator. Serial communication is performed by using a single-line UART that uses the TOOL0 pin.

**Caution The RL78 microcontroller has an on-chip debug function, which is provided for development and evaluation. Do not use the on-chip debug function in products designated for mass production, because the guaranteed number of rewritable times of the flash memory may be exceeded when this function is used, and product reliability therefore cannot be guaranteed. Renesas Electronics is not liable for problems occurring when the on-chip debug function is used.**

<R> **Figure 26-1. Connection Example of E1 On-chip Debugging Emulator**




**Notes 1.** Connecting the dotted line is not necessary during serial flash programming.

**2.** If the reset circuit on the target system does not have a buffer and generates a reset signal only with resistors and capacitors, this pull-up resistor is not necessary.

**Caution This circuit diagram is assumed that the reset signal outputs from an N-ch O.D. buffer (output resistor: 100 Ω or less)**

<R> **Remark** With products not provided with an $EV_{DD0}$ or $EV_{SS0}$ pin, replace $EV_{DD0}$ with $V_{DD}$, or replace $EV_{SS0}$ with $V_{SS}$.

## 26.2 On-Chip Debug Security ID

The RL78 microcontroller has an on-chip debug operation control bit in the flash memory at 000C3H (see **CHAPTER 24 OPTION BYTE**) and an on-chip debug security ID setting area at 000C4H to 000CDH, to prevent third parties from reading memory content.

When the boot swap function is used, also set a value that is the same as that of 010C3H and 010C4H to 010CDH in advance, because 000C3H, 000C4H to 000CDH and 010C3H, and 010C4H to 010CDH are switched.

**Table 26-1. On-Chip Debug Security ID**

| Address | On-Chip Debug Security ID |
|---|---|
| 000C4H to 000CDH | Any ID code of 10 bytes |
| 010C4H to 010CDH | |

## 26.3 Securing of User Resources

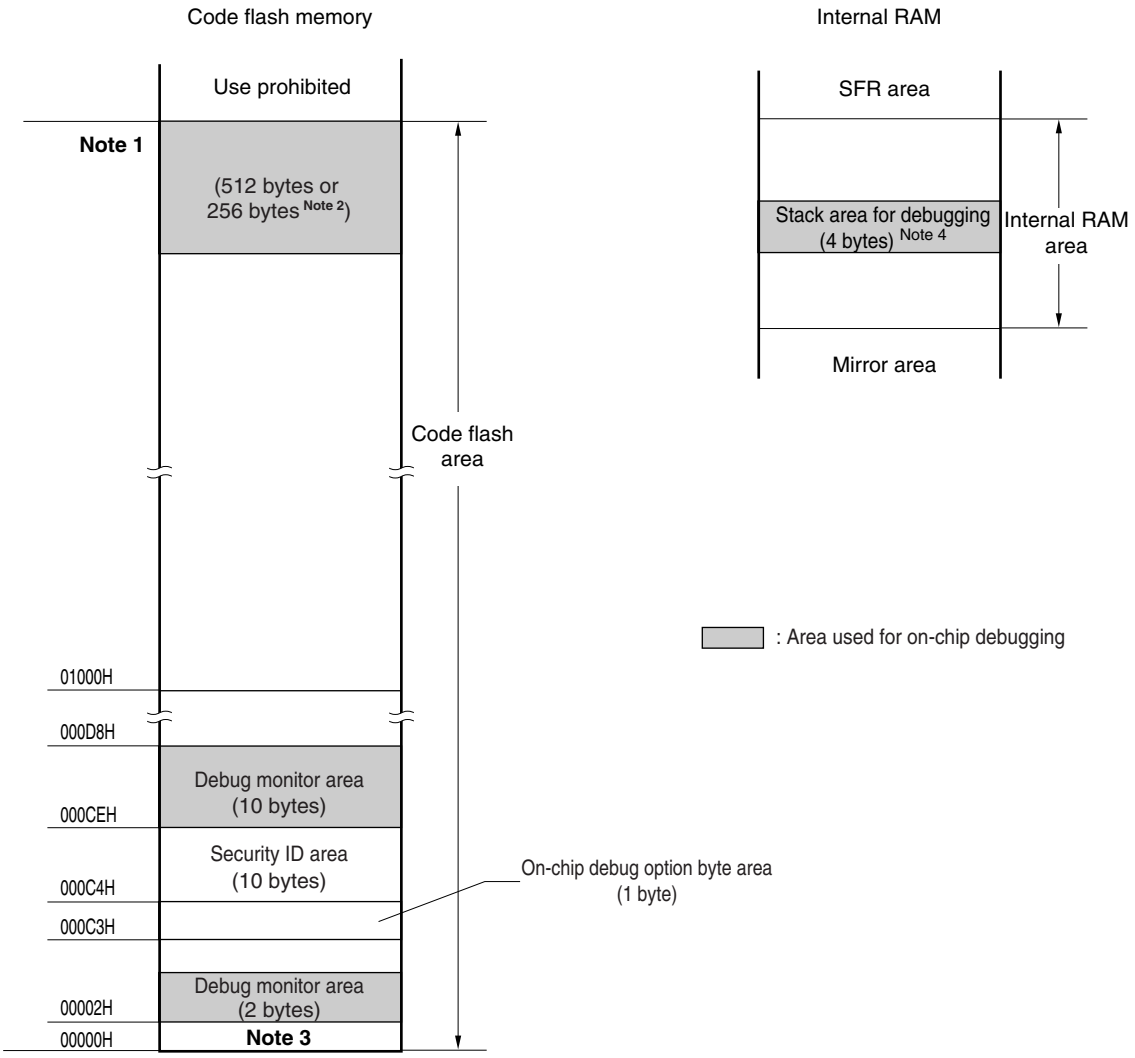
To perform communication between the RL78 microcontroller and E1 on-chip debugging emulator, as well as each debug function, the securing of memory space must be done beforehand.

If Renesas Electronics assembler or compiler is used, the items can be set by using link options.

**(1) Securement of memory space**

The shaded portions in Figure 26-2 are the areas reserved for placing the debug monitor program, so user programs or data cannot be allocated in these spaces. When using the on-chip debug function, these spaces must be secured so as not to be used by the user program. Moreover, this area must not be rewritten by the user program.

**Figure 26-2. Memory Spaces Where Debug Monitor Programs Are Allocated**




**Notes 1.** Address differs depending on products as follows.

| Products (code flash memory capacity) | Address of **Note 1** |
|---|---|
| R5F10ExA (x = 8, B, G) | 03FFFH |
| R5F10ExC (x = 8, B, G, L) | 07FFFH |
| R5F10ExD (x = 8, B, G, L) | 0BFFFH |
| R5F10ExE (x = 8, B, G, L) | 0FFFFH |

**2.** When real-time RAM monitor (RRM) function and dynamic memory modification (DMM) function are not used, it is 256 bytes.

**3.** In debugging, reset vector is rewritten to address allocated to a monitor program.

**4.** Since this area is allocated immediately before the stack area, the address of this area varies depending on the stack increase and decrease. That is, 4 extra bytes are consumed for the stack area used.
When using self-programming, 12 extra bytes are consumed for the stack area used.

# CHAPTER 27 BCD CORRECTION CIRCUIT

## 27.1 BCD Correction Circuit Function

The result of addition/subtraction of the BCD (binary-coded decimal) code and BCD code can be obtained as BCD code with this circuit.

The decimal correction operation result is obtained by performing addition/subtraction having the A register as the operand and then adding/ subtracting the BCD correction result register (BCDADJ).

## 27.2 Registers Used by BCD Correction Circuit

The BCD correction circuit uses the following registers.

- BCD correction result register (BCDADJ)

### 27.2.1 BCD correction result register (BCDADJ)

The BCDADJ register stores correction values for obtaining the add/subtract result as BCD code through add/subtract instructions using the A register as the operand.

The value read from the BCDADJ register varies depending on the value of the A register when it is read and those of the CY and AC flags.

The BCDADJ register is read by an 8-bit memory manipulation instruction.

Reset input sets this register to undefined.

**Figure 27-1. Format of BCD Correction Result Register (BCDADJ)**

Address: F00FEH After reset: undefined R

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| BCDADJ | | | | | | | | |

## 27.3 BCD Correction Circuit Operation

The basic operation of the BCD correction circuit is as follows.

**(1) Addition: Calculating the result of adding a BCD code value and another BCD code value by using a BCD code value**

<1> The BCD code value to which addition is performed is stored in the A register.

<2> By adding the value of the A register and the second operand (value of one more BCD code to be added) as are in binary, the binary operation result is stored in the A register and the correction value is stored in the BCD correction result register (BCDADJ).

<3> Decimal correction is performed by adding in binary the value of the A register (addition result in binary) and the BCDADJ register (correction value), and the correction result is stored in the A register and CY flag.

**Caution The value read from the BCDADJ register varies depending on the value of the A register when it is read and those of the CY and AC flags. Therefore, execute the instruction <3> after the instruction <2> instead of executing any other instructions. To perform BCD correction in the interrupt enabled state, saving and restoring the A register is required within the interrupt function. PSW (CY flag and AC flag) is restored by the RETI instruction.**

An example is shown below.

Examples 1: 99 + 89 = 188

| Instruction | A Register | CY Flag | AC Flag | BCDADJ Register |
|---|---|---|---|---|
| `MOV A, #99H ; <1>` | 99H | – | – | – |
| `ADD A, #89H ; <2>` | 22H | 1 | 1 | 66H |
| `ADD A, !BCDADJ ; <3>` | 88H | 1 | 0 | – |

Examples 2: 85 + 15 = 100

| Instruction | A Register | CY Flag | AC Flag | BCDADJ Register |
|---|---|---|---|---|
| `MOV A, #85H ; <1>` | 85H | – | – | – |
| `ADD A, #15H ; <2>` | 9AH | 0 | 0 | 66H |
| `ADD A, !BCDADJ ; <3>` | 00H | 1 | 1 | – |

Examples 3: 80 + 80 = 160

| Instruction | A Register | CY Flag | AC Flag | BCDADJ Register |
|---|---|---|---|---|
| `MOV A, #80H ; <1>` | 80H | – | – | – |
| `ADD A, #80H ; <2>` | 00H | 1 | 0 | 60H |
| `ADD A, !BCDADJ ; <3>` | 60H | 1 | 0 | – |

**(2) Subtraction: Calculating the result of subtracting a BCD code value from another BCD code value by using a BCD code value**

<1> The BCD code value from which subtraction is performed is stored in the A register.

<2> By subtracting the value of the second operand (value of BCD code to be subtracted) from the A register as is in binary, the calculation result in binary is stored in the A register, and the correction value is stored in the BCD correction result register (BCDADJ).

<3> Decimal correction is performed by subtracting the value of the BCDADJ register (correction value) from the A register (subtraction result in binary) in binary, and the correction result is stored in the A register and CY flag.

**Caution The value read from the BCDADJ register varies depending on the value of the A register when it is read and those of the CY and AC flags. Therefore, execute the instruction <3> after the instruction <2> instead of executing any other instructions. To perform BCD correction in the interrupt enabled state, saving and restoring the A register is required within the interrupt function. PSW (CY flag and AC flag) is restored by the RETI instruction.**

An example is shown below.

Example: 91 − 52 = 39

| Instruction | A Register | CY Flag | AC Flag | BCDADJ Register |
|---|---|---|---|---|
| `MOV A, #91H ; <1>` | 91H | − | − | − |
| `SUB A, #52H ; <2>` | 3FH | 0 | 1 | 06H |
| `SUB A, !BCDADJ ; <3>` | 39H | 0 | 0 | − |

# CHAPTER 28 INSTRUCTION SET

<R> This chapter lists the instructions in the RL78 microcontroller instruction set. For details of each operation and operation code, refer to the separate document **RL78 Microcontrollers User's Manual: Software (R01US0015).**

## 28.1 Conventions Used in Operation List

### 28.1.1 Operand identifiers and specification methods

Operands are described in the "Operand" column of each instruction in accordance with the description method of the instruction operand identifier (refer to the assembler specifications for details). When there are two or more description methods, select one of them. Alphabetic letters in capitals and the symbols, #, !, !!, $, $!, [ ], and ES: are keywords and are described as they are. Each symbol has the following meaning.

- #: Immediate data specification
- !: 16-bit absolute address specification
- !!: 20-bit absolute address specification
- $: 8-bit relative address specification
- $!: 16-bit relative address specification
- [ ]: Indirect address specification
- ES:: Extension address specification

In the case of immediate data, describe an appropriate numeric value or a label. When using a label, be sure to describe the #, !, !!, $, $!, [ ], and ES: symbols.

For operand register identifiers, r and rp, either function names (X, A, C, etc.) or absolute names (names in parentheses in the table below, R0, R1, R2, etc.) can be used for description.

**Table 28-1. Operand Identifiers and Specification Methods**

| Identifier | Description Method |
|---|---|
| r<br>rp<br>sfr<br>sfrp | X (R0), A (R1), C (R2), B (R3), E (R4), D (R5), L (R6), H (R7)<br>AX (RP0), BC (RP1), DE (RP2), HL (RP3)<br>Special-function register symbol (SFR symbol) FFF00H to FFFFFH<br>Special-function register symbols (16-bit manipulatable SFR symbol. Even addresses only[Note]) FFF00H to FFFFFH |
| saddr<br>saddrp | FFE20H to FFF1FH Immediate data or labels<br>FFE20H to FF1FH Immediate data or labels (even addresses only[Note]) |
| addr20<br>addr16<br>addr5 | 00000H to FFFFFH Immediate data or labels<br>0000H to FFFFH Immediate data or labels (only even addresses for 16-bit data transfer instructions[Note])<br>0080H to 00BFH Immediate data or labels (even addresses only[Note]) |
| word<br>byte<br>bit | 16-bit immediate data or label<br>8-bit immediate data or label<br>3-bit immediate data or label |
| RBn | RB0 to RB3 |

**Note** Bit 0 = 0 when an odd address is specified.

**Remark** The special function registers can be described to operand sfr as symbols. See **Table 3-5 SFR List** for the symbols of the special function registers. The extended special function registers can be described to operand !addr16 as symbols. See **Table 3-6 Extended SFR (2nd SFR) List** for the symbols of the extended special function registers.

### 28.1.2 Description of operation column

The operation when the instruction is executed is shown in the “Operation” column using the following symbols.

**Table 28-2. Symbols in “Operation” Column**

| Symbol | Function |
|---|---|
| A | A register; 8-bit accumulator |
| X | X register |
| B | B register |
| C | C register |
| D | D register |
| E | E register |
| H | H register |
| L | L register |
| ES | ES register |
| CS | CS register |
| AX | AX register pair; 16-bit accumulator |
| BC | BC register pair |
| DE | DE register pair |
| HL | HL register pair |
| PC | Program counter |
| SP | Stack pointer |
| PSW | Program status word |
| CY | Carry flag |
| AC | Auxiliary carry flag |
| Z | Zero flag |
| RBS | Register bank select flag |
| IE | Interrupt request enable flag |
| () | Memory contents indicated by address or register contents in parentheses |
| $X_H$, $X_L$<br>$X_S$, $X_H$, $X_L$ | 16-bit registers: $X_H$ = higher 8 bits, $X_L$ = lower 8 bits<br>20-bit registers: $X_S$ = (bits 19 to 16), $X_H$ = (bits 15 to 8), $X_L$ = (bits 7 to 0) |
| ∧ | Logical product (AND) |
| ∨ | Logical sum (OR) |
| ∀ | Exclusive logical sum (exclusive OR) |
| ¯ | Inverted data |
| addr5 | 16-bit immediate data (even addresses only in 0080H to 00BFH) |
| addr16 | 16-bit immediate data |
| addr20 | 20-bit immediate data |
| jdisp8 | Signed 8-bit data (displacement value) |
| jdisp16 | Signed 16-bit data (displacement value) |

### 28.1.3 Description of flag operation column

The change of the flag value when the instruction is executed is shown in the “Flag” column using the following symbols.

**Table 28-3. Symbols in “Flag” Column**

| Symbol | Change of Flag Value |
|---|---|
| (Blank) | Unchanged |
| 0 | Cleared to 0 |
| 1 | Set to 1 |
| × | Set/cleared according to the result |
| R | Previously saved value is restored |

### 28.1.4 PREFIX instruction

Instructions with “ES:” have a PREFIX operation code as a prefix to extend the accessible data area to the 1 MB space (00000H to FFFFFH), by adding the ES register value to the 64 KB space from F0000H to FFFFFH. When a PREFIX operation code is attached as a prefix to the target instruction, only one instruction immediately after the PREFIX operation code is executed as the addresses with the ES register value added.

A interrupt and DMA transfer are not acknowledged between a PREFIX instruction code and the instruction immediately after.

**Table 28-4. Use Example of PREFIX Operation Code**

| Instruction | Opcode | | | | |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 |
| MOV !addr16, #byte | CFH | !addr16 | | #byte | – |
| MOV ES:!addr16, #byte | 11H | CFH | !addr16 | | #byte |
| MOV A, [HL] | 8BH | – | – | – | – |
| MOV A, ES:[HL] | 11H | 8BH | – | – | – |

**Caution Set the ES register value with MOV ES, A, etc., before executing the PREFIX instruction.**

## 28.2 Operation List

**Table 28-5. Operation List (1/17)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | Flag | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | Note 1 | Note 2 | | Z | AC | CY |
| 8-bit data transfer | MOV | r, #byte | 2 | 1 | – | r ← byte | | | |
| | | PSW, #byte | 3 | 3 | – | PSW ← byte | × | × | × |
| | | CS, #byte | 3 | 1 | – | CS ← byte | | | |
| | | ES, #byte | 2 | 1 | – | ES ← byte | | | |
| | | !addr16, #byte | 4 | 1 | – | (addr16) ← byte | | | |
| | | ES:!addr16, #byte | 5 | 2 | – | (ES, addr16) ← byte | | | |
| | | saddr, #byte | 3 | 1 | – | (saddr) ← byte | | | |
| | | sfr, #byte | 3 | 1 | – | sfr ← byte | | | |
| | | [DE+byte], #byte | 3 | 1 | – | (DE+byte) ← byte | | | |
| | | ES:[DE+byte],#byte | 4 | 2 | – | ((ES, DE)+byte) ← byte | | | |
| | | [HL+byte], #byte | 3 | 1 | – | (HL+byte) ← byte | | | |
| | | ES:[HL+byte],#byte | 4 | 2 | – | ((ES, HL)+byte) ← byte | | | |
| | | [SP+byte], #byte | 3 | 1 | – | (SP+byte) ← byte | | | |
| | | word[B], #byte | 4 | 1 | – | (B+word) ← byte | | | |
| | | ES:word[B], #byte | 5 | 2 | – | ((ES, B)+word) ← byte | | | |
| | | word[C], #byte | 4 | 1 | – | (C+word) ← byte | | | |
| | | ES:word[C], #byte | 5 | 2 | – | ((ES, C)+word) ← byte | | | |
| | | word[BC], #byte | 4 | 1 | – | (BC+word) ← byte | | | |
| | | ES:word[BC], #byte | 5 | 2 | – | ((ES, BC)+word) ← byte | | | |
| | | A, r Note 3 | 1 | 1 | – | A ← r | | | |
| | | r, A Note 3 | 1 | 1 | – | r ← A | | | |
| | | A, PSW | 2 | 1 | – | A ← PSW | | | |
| | | PSW, A | 2 | 3 | – | PSW ← A | × | × | × |
| | | A, CS | 2 | 1 | – | A ← CS | | | |
| | | CS, A | 2 | 1 | – | CS ← A | | | |
| | | A, ES | 2 | 1 | – | A ← ES | | | |
| | | ES, A | 2 | 1 | – | ES ← A | | | |
| | | A, !addr16 | 3 | 1 | 4 | A ← (addr16) | | | |
| | | A, ES:!addr16 | 4 | 2 | 5 | A ← (ES, addr16) | | | |
| | | !addr16, A | 3 | 1 | – | (addr16) ← A | | | |
| | | ES:!addr16, A | 4 | 2 | – | (ES, addr16) ← A | | | |
| | | A, saddr | 2 | 1 | – | A ← (saddr) | | | |
| | | saddr, A | 2 | 1 | – | (saddr) ← A | | | |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

<R> **2.** Number of CPU clocks ($f_{CLK}$) when the code flash memory is accessed, or when the data flash memory is accessed by an 8-bit instruction.

**3.** Except r = A

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 28-5. Operation List (2/17)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | Flag |
|---|---|---|---|---|---|---|---|
| | | | | Note 1 | Note 2 | | Z AC CY |
| 8-bit data transfer | MOV | A, sfr | 2 | 1 | – | A ← sfr | |
| | | sfr, A | 2 | 1 | – | sfr ← A | |
| | | A, [DE] | 1 | 1 | 4 | A ← (DE) | |
| | | [DE], A | 1 | 1 | – | (DE) ← A | |
| | | A, ES:[DE] | 2 | 2 | 5 | A ← (ES, DE) | |
| | | ES:[DE], A | 2 | 2 | – | (ES, DE) ← A | |
| | | A, [HL] | 1 | 1 | 4 | A ← (HL) | |
| | | [HL], A | 1 | 1 | – | (HL) ← A | |
| | | A, ES:[HL] | 2 | 2 | 5 | A ← (ES, HL) | |
| | | ES:[HL], A | 2 | 2 | – | (ES, HL) ← A | |
| | | A, [DE+byte] | 2 | 1 | 4 | A ← (DE + byte) | |
| | | [DE+byte], A | 2 | 1 | – | (DE + byte) ← A | |
| | | A, ES:[DE+byte] | 3 | 2 | 5 | A ← ((ES, DE) + byte) | |
| | | ES:[DE+byte], A | 3 | 2 | – | ((ES, DE) + byte) ← A | |
| | | A, [HL+byte] | 2 | 1 | 4 | A ← (HL + byte) | |
| | | [HL+byte], A | 2 | 1 | – | (HL + byte) ← A | |
| | | A, ES:[HL+byte] | 3 | 2 | 5 | A ← ((ES, HL) + byte) | |
| | | ES:[HL+byte], A | 3 | 2 | – | ((ES, HL) + byte) ← A | |
| | | A, [SP+byte] | 2 | 1 | – | A ← (SP + byte) | |
| | | [SP+byte], A | 2 | 1 | – | (SP + byte) ← A | |
| | | A, word[B] | 3 | 1 | 4 | A ← (B + word) | |
| | | word[B], A | 3 | 1 | – | (B + word) ← A | |
| | | A, ES:word[B] | 4 | 2 | 5 | A ← ((ES, B) + word) | |
| | | ES:word[B], A | 4 | 2 | – | ((ES, B) + word) ← A | |
| | | A, word[C] | 3 | 1 | 4 | A ← (C + word) | |
| | | word[C], A | 3 | 1 | – | (C + word) ← A | |
| | | A, ES:word[C] | 4 | 2 | 5 | A ← ((ES, C) + word) | |
| | | ES:word[C], A | 4 | 2 | – | ((ES, C) + word) ← A | |
| | | A, word[BC] | 3 | 1 | 4 | A ← (BC + word) | |
| | | word[BC], A | 3 | 1 | – | (BC + word) ← A | |
| | | A, ES:word[BC] | 4 | 2 | 5 | A ← ((ES, BC) + word) | |
| | | ES:word[BC], A | 4 | 2 | – | ((ES, BC) + word) ← A | |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

<R> **2.** Number of CPU clocks ($f_{CLK}$) when the code flash memory is accessed, or when the data flash memory is accessed by an 8-bit instruction.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 28-5. Operation List (3/17)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | Flag |
|---|---|---|---|---|---|---|---|
| | | | | Note 1 | Note 2 | | Z AC CY |
| 8-bit data transfer | MOV | A, [HL+B] | 2 | 1 | 4 | A ← (HL + B) | |
| | | [HL+B], A | 2 | 1 | – | (HL + B) ← A | |
| | | A, ES:[HL+B] | 3 | 2 | 5 | A ← ((ES, HL) + B) | |
| | | ES:[HL+B], A | 3 | 2 | – | ((ES, HL) + B) ← A | |
| | | A, [HL+C] | 2 | 1 | 4 | A ← (HL + C) | |
| | | [HL+C], A | 2 | 1 | – | (HL + C) ← A | |
| | | A, ES:[HL+C] | 3 | 2 | 5 | A ← ((ES, HL) + C) | |
| | | ES:[HL+C], A | 3 | 2 | – | ((ES, HL) + C) ← A | |
| | | X, !addr16 | 3 | 1 | 4 | X ← (addr16) | |
| | | X, ES:!addr16 | 4 | 2 | 5 | X ← (ES, addr16) | |
| | | X, saddr | 2 | 1 | – | X ← (saddr) | |
| | | B, !addr16 | 3 | 1 | 4 | B ← (addr16) | |
| | | B, ES:!addr16 | 4 | 2 | 5 | B ← (ES, addr16) | |
| | | B, saddr | 2 | 1 | – | B ← (saddr) | |
| | | C, !addr16 | 3 | 1 | 4 | C ← (addr16) | |
| | | C, ES:!addr16 | 4 | 2 | 5 | C ← (ES, addr16) | |
| | | C, saddr | 2 | 1 | – | C ← (saddr) | |
| | | ES, saddr | 3 | 1 | – | ES ← (saddr) | |
| | XCH | A, r Note 3 | 1 (r = X)<br>2 (other than r = X) | 1 | – | A ←→ r | |
| | | A, !addr16 | 4 | 2 | – | A ←→ (addr16) | |
| | | A, ES:!addr16 | 5 | 3 | – | A ←→ (ES, addr16) | |
| | | A, saddr | 3 | 2 | – | A ←→ (saddr) | |
| | | A, sfr | 3 | 2 | – | A ←→ sfr | |
| | | A, [DE] | 2 | 2 | – | A ←→ (DE) | |
| | | A, ES:[DE] | 3 | 3 | – | A ←→ (ES, DE) | |
| | | A, [HL] | 2 | 2 | – | A ←→ (HL) | |
| | | A, ES:[HL] | 3 | 3 | – | A ←→ (ES, HL) | |
| | | A, [DE+byte] | 3 | 2 | – | A ←→ (DE + byte) | |
| | | A, ES:[DE+byte] | 4 | 3 | – | A ←→ ((ES, DE) + byte) | |
| | | A, [HL+byte] | 3 | 2 | – | A ←→ (HL + byte) | |
| | | A, ES:[HL+byte] | 4 | 3 | – | A ←→ ((ES, HL) + byte) | |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

<R> **2.** Number of CPU clocks ($f_{CLK}$) when the code flash memory is accessed, or when the data flash memory is accessed by an 8-bit instruction.

**3.** Except r = A

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 28-5. Operation List (4/17)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | Flag |
|---|---|---|---|---|---|---|---|
| | | | | Note 1 | Note 2 | | Z AC CY |
| 8-bit data transfer | XCH | A, [HL+B] | 2 | 2 | – | A ←→ (HL+B) | |
| | | A, ES:[HL+B] | 3 | 3 | – | A ←→ ((ES, HL)+B) | |
| | | A, [HL+C] | 2 | 2 | – | A ←→ (HL+C) | |
| | | A, ES:[HL+C] | 3 | 3 | – | A ←→ ((ES, HL)+C) | |
| | ONEB | A | 1 | 1 | – | A ← 01H | |
| | | X | 1 | 1 | – | X ← 01H | |
| | | B | 1 | 1 | – | B ← 01H | |
| | | C | 1 | 1 | – | C ← 01H | |
| | | !addr16 | 3 | 1 | – | (addr16) ← 01H | |
| | | ES:!addr16 | 4 | 2 | – | (ES, addr16) ← 01H | |
| | | saddr | 2 | 1 | – | (saddr) ← 01H | |
| | CLRB | A | 1 | 1 | – | A ← 00H | |
| | | X | 1 | 1 | – | X ← 00H | |
| | | B | 1 | 1 | – | B ← 00H | |
| | | C | 1 | 1 | – | C ← 00H | |
| | | !addr16 | 3 | 1 | – | (addr16) ← 00H | |
| | | ES:!addr16 | 4 | 2 | – | (ES,addr16) ← 00H | |
| | | saddr | 2 | 1 | – | (saddr) ← 00H | |
| | MOVS | [HL+byte], X | 3 | 1 | – | (HL+byte) ← X | × × |
| | | ES:[HL+byte], X | 4 | 2 | – | (ES, HL+byte) ← X | × × |
| 16-bit data transfer | MOVW | rp, #word | 3 | 1 | – | rp ← word | |
| | | saddrp, #word | 4 | 1 | – | (saddrp) ← word | |
| | | sfrp, #word | 4 | 1 | – | sfrp ← word | |
| | | AX, rp Note 3 | 1 | 1 | – | AX ← rp | |
| | | rp, AX Note 3 | 1 | 1 | – | rp ← AX | |
| | | AX, !addr16 | 3 | 1 | 4 | AX ← (addr16) | |
| | | !addr16, AX | 3 | 1 | – | (addr16) ← AX | |
| | | AX, ES:!addr16 | 4 | 2 | 5 | AX ← (ES, addr16) | |
| | | ES:!addr16, AX | 4 | 2 | – | (ES, addr16) ← AX | |
| | | AX, saddrp | 2 | 1 | – | AX ← (saddrp) | |
| | | saddrp, AX | 2 | 1 | – | (saddrp) ← AX | |
| | | AX, sfrp | 2 | 1 | – | AX ← sfrp | |
| | | sfrp, AX | 2 | 1 | – | sfrp ← AX | |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

<R> **2.** Number of CPU clocks ($f_{CLK}$) when the code flash memory is accessed, or when the data flash memory is accessed by an 8-bit instruction.

**3.** Except rp = AX

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 28-5. Operation List (5/17)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | Flag |
|---|---|---|---|---|---|---|---|
| | | | | Note 1 | Note 2 | | Z AC CY |
| 16-bit data transfer | MOVW | AX, [DE] | 1 | 1 | 4 | AX ← (DE) | |
| | | [DE], AX | 1 | 1 | – | (DE) ← AX | |
| | | AX, ES:[DE] | 2 | 2 | 5 | AX ← (ES, DE) | |
| | | ES:[DE], AX | 2 | 2 | – | (ES, DE) ← AX | |
| | | AX, [HL] | 1 | 1 | 4 | AX ← (HL) | |
| | | [HL], AX | 1 | 1 | – | (HL) ← AX | |
| | | AX, ES:[HL] | 2 | 2 | 5 | AX ← (ES, HL) | |
| | | ES:[HL], AX | 2 | 2 | – | (ES, HL) ← AX | |
| | | AX, [DE+byte] | 2 | 1 | 4 | AX ← (DE+byte) | |
| | | [DE+byte], AX | 2 | 1 | – | (DE+byte) ← AX | |
| | | AX, ES:[DE+byte] | 3 | 2 | 5 | AX ← ((ES, DE) + byte) | |
| | | ES:[DE+byte], AX | 3 | 2 | – | ((ES, DE) + byte) ← AX | |
| | | AX, [HL+byte] | 2 | 1 | 4 | AX ← (HL + byte) | |
| | | [HL+byte], AX | 2 | 1 | – | (HL + byte) ← AX | |
| | | AX, ES:[HL+byte] | 3 | 2 | 5 | AX ← ((ES, HL) + byte) | |
| | | ES:[HL+byte], AX | 3 | 2 | – | ((ES, HL) + byte) ← AX | |
| | | AX, [SP+byte] | 2 | 1 | – | AX ← (SP + byte) | |
| | | [SP+byte], AX | 2 | 1 | – | (SP + byte) ← AX | |
| | | AX, word[B] | 3 | 1 | 4 | AX ← (B + word) | |
| | | word[B], AX | 3 | 1 | – | (B+ word) ← AX | |
| | | AX, ES:word[B] | 4 | 2 | 5 | AX ← ((ES, B) + word) | |
| | | ES:word[B], AX | 4 | 2 | – | ((ES, B) + word) ← AX | |
| | | AX, word[C] | 3 | 1 | 4 | AX ← (C + word) | |
| | | word[C], AX | 3 | 1 | – | (C + word) ← AX | |
| | | AX, ES:word[C] | 4 | 2 | 5 | AX ← ((ES, C) + word) | |
| | | ES:word[C], AX | 4 | 2 | – | ((ES, C) + word) ← AX | |
| | | AX, word[BC] | 3 | 1 | 4 | AX ← (BC + word) | |
| | | word[BC], AX | 3 | 1 | – | (BC + word) ← AX | |
| | | AX, ES:word[BC] | 4 | 2 | 5 | AX ← ((ES, BC) + word) | |
| | | ES:word[BC], AX | 4 | 2 | – | ((ES, BC) + word) ← AX | |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

<R> **2.** Number of CPU clocks ($f_{CLK}$) when the code flash memory is accessed, or when the data flash memory is accessed by an 8-bit instruction.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 28-5. Operation List (6/17)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | Flag | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | Note 1 | Note 2 | | Z | AC | CY |
| 16-bit data transfer | MOVW | BC, !addr16 | 3 | 1 | 4 | BC ← (addr16) | | | |
| | | BC, ES:!addr16 | 4 | 2 | 5 | BC ← (ES, addr16) | | | |
| | | DE, !addr16 | 3 | 1 | 4 | DE ← (addr16) | | | |
| | | DE, ES:!addr16 | 4 | 2 | 5 | DE ← (ES, addr16) | | | |
| | | HL, !addr16 | 3 | 1 | 4 | HL ← (addr16) | | | |
| | | HL, ES:!addr16 | 4 | 2 | 5 | HL ← (ES, addr16) | | | |
| | | BC, saddrp | 2 | 1 | – | BC ← (saddrp) | | | |
| | | DE, saddrp | 2 | 1 | – | DE ← (saddrp) | | | |
| | | HL, saddrp | 2 | 1 | – | HL ← (saddrp) | | | |
| | XCHW | AX, rp Note 3 | 1 | 1 | – | AX ←→ rp | | | |
| | ONEW | AX | 1 | 1 | – | AX ← 0001H | | | |
| | | BC | 1 | 1 | – | BC ← 0001H | | | |
| | CLRW | AX | 1 | 1 | – | AX ← 0000H | | | |
| | | BC | 1 | 1 | – | BC ← 0000H | | | |
| 8-bit operation | ADD | A, #byte | 2 | 1 | – | A, CY ← A + byte | × | × | × |
| | | saddr, #byte | 3 | 2 | – | (saddr), CY ← (saddr)+byte | × | × | × |
| | | A, r Note 4 | 2 | 1 | – | A, CY ← A + r | × | × | × |
| | | r, A | 2 | 1 | – | r, CY ← r + A | × | × | × |
| | | A, !addr16 | 3 | 1 | 4 | A, CY ← A + (addr16) | × | × | × |
| | | A, ES:!addr16 | 4 | 2 | 5 | A, CY ← A + (ES, addr16) | × | × | × |
| | | A, saddr | 2 | 1 | – | A, CY ← A + (saddr) | × | × | × |
| | | A, [HL] | 1 | 1 | 4 | A, CY ← A+ (HL) | × | × | × |
| | | A, ES:[HL] | 2 | 2 | 5 | A,CY ← A + (ES, HL) | × | × | × |
| | | A, [HL+byte] | 2 | 1 | 4 | A, CY ← A + (HL+byte) | × | × | × |
| | | A, ES:[HL+byte] | 3 | 2 | 5 | A,CY ← A + ((ES, HL)+byte) | × | × | × |
| | | A, [HL+B] | 2 | 1 | 4 | A, CY ← A + (HL+B) | × | × | × |
| | | A, ES:[HL+B] | 3 | 2 | 5 | A,CY ← A+((ES, HL)+B) | × | × | × |
| | | A, [HL+C] | 2 | 1 | 4 | A, CY ← A + (HL+C) | × | × | × |
| | | A, ES:[HL+C] | 3 | 2 | 5 | A,CY ← A + ((ES, HL) + C) | × | × | × |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

<R> **2.** Number of CPU clocks ($f_{CLK}$) when the code flash memory is accessed, or when the data flash memory is accessed by an 8-bit instruction.

**3.** Except rp = AX

**4.** Except r = A

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 28-5. Operation List (7/17)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | Flag | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | Note 1 | Note 2 | | Z | AC | CY |
| 8-bit operation | ADDC | A, #byte | 2 | 1 | – | A, CY ← A+byte+CY | × | × | × |
| | | saddr, #byte | 3 | 2 | – | (saddr), CY ← (saddr) +byte+CY | × | × | × |
| | | A, rv Note 3 | 2 | 1 | – | A, CY ← A + r + CY | × | × | × |
| | | r, A | 2 | 1 | – | r, CY ← r + A + CY | × | × | × |
| | | A, !addr16 | 3 | 1 | 4 | A, CY ← A + (addr16)+CY | × | × | × |
| | | A, ES:!addr16 | 4 | 2 | 5 | A, CY ← A + (ES, addr16)+CY | × | × | × |
| | | A, saddr | 2 | 1 | – | A, CY ← A + (saddr)+CY | × | × | × |
| | | A, [HL] | 1 | 1 | 4 | A, CY ← A+ (HL) + CY | × | × | × |
| | | A, ES:[HL] | 2 | 2 | 5 | A,CY ← A+ (ES, HL) + CY | × | × | × |
| | | A, [HL+byte] | 2 | 1 | 4 | A, CY ← A+ (HL+byte) + CY | × | × | × |
| | | A, ES:[HL+byte] | 3 | 2 | 5 | A,CY ← A+ ((ES, HL)+byte) + CY | × | × | × |
| | | A, [HL+B] | 2 | 1 | 4 | A, CY ← A+ (HL+B) +CY | × | × | × |
| | | A, ES:[HL+B] | 3 | 2 | 5 | A,CY ← A+((ES, HL)+B)+CY | × | × | × |
| | | A, [HL+C] | 2 | 1 | 4 | A, CY ← A+ (HL+C)+CY | × | × | × |
| | | A, ES:[HL+C] | 3 | 2 | 5 | A,CY ← A+ ((ES, HL)+C)+CY | × | × | × |
| | SUB | A, #byte | 2 | 1 | – | A, CY ← A – byte | × | × | × |
| | | saddr, #byte | 3 | 2 | – | (saddr), CY ← (saddr) – byte | × | × | × |
| | | A, r Note 3 | 2 | 1 | – | A, CY ← A – r | × | × | × |
| | | r, A | 2 | 1 | – | r, CY ← r – A | × | × | × |
| | | A, !addr16 | 3 | 1 | 4 | A, CY ← A – (addr16) | × | × | × |
| | | A, ES:!addr16 | 4 | 2 | 5 | A, CY ← A – (ES, addr16) | × | × | × |
| | | A, saddr | 2 | 1 | – | A, CY ← A – (saddr) | × | × | × |
| | | A, [HL] | 1 | 1 | 4 | A, CY ← A – (HL) | × | × | × |
| | | A, ES:[HL] | 2 | 2 | 5 | A,CY ← A – (ES, HL) | × | × | × |
| | | A, [HL+byte] | 2 | 1 | 4 | A, CY ← A – (HL+byte) | × | × | × |
| | | A, ES:[HL+byte] | 3 | 2 | 5 | A,CY ← A – ((ES, HL)+byte) | × | × | × |
| | | A, [HL+B] | 2 | 1 | 4 | A, CY ← A – (HL+B) | × | × | × |
| | | A, ES:[HL+B] | 3 | 2 | 5 | A,CY ← A – ((ES, HL)+B) | × | × | × |
| | | A, [HL+C] | 2 | 1 | 4 | A, CY ← A – (HL+C) | × | × | × |
| | | A, ES:[HL+C] | 3 | 2 | 5 | A,CY ← A – ((ES, HL)+C) | × | × | × |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

<R> **2.** Number of CPU clocks ($f_{CLK}$) when the code flash memory is accessed, or when the data flash memory is accessed by an 8-bit instruction.

**3.** Except r = A

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 28-5. Operation List (8/17)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | Flag | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | Note 1 | Note 2 | | Z | AC | CY |
| 8-bit operation | SUBC | A, #byte | 2 | 1 | – | A, CY ← A – byte – CY | × | × | × |
| | | saddr, #byte | 3 | 2 | – | (saddr), CY ← (saddr) – byte – CY | × | × | × |
| | | A, r Note 3 | 2 | 1 | – | A, CY ← A – r – CY | × | × | × |
| | | r, A | 2 | 1 | – | r, CY ← r – A – CY | × | × | × |
| | | A, !addr16 | 3 | 1 | 4 | A, CY ← A – (addr16) – CY | × | × | × |
| | | A, ES:!addr16 | 4 | 2 | 5 | A, CY ← A – (ES, addr16) – CY | × | × | × |
| | | A, saddr | 2 | 1 | – | A, CY ← A – (saddr) – CY | × | × | × |
| | | A, [HL] | 1 | 1 | 4 | A, CY ← A – (HL) – CY | × | × | × |
| | | A, ES:[HL] | 2 | 2 | 5 | A,CY ← A – (ES, HL) – CY | × | × | × |
| | | A, [HL+byte] | 2 | 1 | 4 | A, CY ← A – (HL+byte) – CY | × | × | × |
| | | A, ES:[HL+byte] | 3 | 2 | 5 | A,CY ← A – ((ES, HL)+byte) – CY | × | × | × |
| | | A, [HL+B] | 2 | 1 | 4 | A, CY ← A – (HL+B) – CY | × | × | × |
| | | A, ES:[HL+B] | 3 | 2 | 5 | A,CY ← A – ((ES, HL)+B) – CY | × | × | × |
| | | A, [HL+C] | 2 | 1 | 4 | A, CY ← A – (HL+C) – CY | × | × | × |
| | | A, ES:[HL+C] | 3 | 2 | 5 | A, CY ← A – ((ES:HL)+C) – CY | × | × | × |
| | AND | A, #byte | 2 | 1 | – | A ← A ∧ byte | × | | |
| | | saddr, #byte | 3 | 2 | – | (saddr) ← (saddr) ∧ byte | × | | |
| | | A, r Note 3 | 2 | 1 | – | A ← A ∧ r | × | | |
| | | r, A | 2 | 1 | – | R ← r ∧ A | × | | |
| | | A, !addr16 | 3 | 1 | 4 | A ← A ∧ (addr16) | × | | |
| | | A, ES:!addr16 | 4 | 2 | 5 | A ← A ∧ (ES:addr16) | × | | |
| | | A, saddr | 2 | 1 | – | A ← A ∧ (saddr) | × | | |
| | | A, [HL] | 1 | 1 | 4 | A ← A ∧ (HL) | × | | |
| | | A, ES:[HL] | 2 | 2 | 5 | A ← A ∧ (ES:HL) | × | | |
| | | A, [HL+byte] | 2 | 1 | 4 | A ← A ∧ (HL+byte) | × | | |
| | | A, ES:[HL+byte] | 3 | 2 | 5 | A ← A ∧ ((ES:HL)+byte) | × | | |
| | | A, [HL+B] | 2 | 1 | 4 | A ← A ∧ (HL+B) | × | | |
| | | A, ES:[HL+B] | 3 | 2 | 5 | A ← A ∧ ((ES:HL)+B) | × | | |
| | | A, [HL+C] | 2 | 1 | 4 | A ← A ∧ (HL+C) | × | | |
| | | A, ES:[HL+C] | 3 | 2 | 5 | A ← A ∧ ((ES:HL)+C) | × | | |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

<R> **2.** Number of CPU clocks ($f_{CLK}$) when the code flash memory is accessed, or when the data flash memory is accessed by an 8-bit instruction.

**3.** Except r = A

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 28-5. Operation List (9/17)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | Flag |
|---|---|---|---|---|---|---|---|
| | | | | Note 1 | Note 2 | | Z AC CY |
| 8-bit operation | OR | A, #byte | 2 | 1 | – | A ← A∨byte | × |
| | | saddr, #byte | 3 | 2 | – | (saddr) ← (saddr)∨byte | × |
| | | A, r Note 3 | 2 | 1 | – | A ← A∨r | × |
| | | r, A | 2 | 1 | – | r ← r∨A | × |
| | | A, !addr16 | 3 | 1 | 4 | A ← A∨(addr16) | × |
| | | A, ES:!addr16 | 4 | 2 | 5 | A ← A∨(ES:addr16) | × |
| | | A, saddr | 2 | 1 | – | A ← A∨(saddr) | × |
| | | A, [HL] | 1 | 1 | 4 | A ← A∨(H) | × |
| | | A, ES:[HL] | 2 | 2 | 5 | A ← A∨(ES:HL) | × |
| | | A, [HL+byte] | 2 | 1 | 4 | A ← A∨(HL+byte) | × |
| | | A, ES:[HL+byte] | 3 | 2 | 5 | A ← A∨((ES:HL)+byte) | × |
| | | A, [HL+B] | 2 | 1 | 4 | A ← A∨(HL+B) | × |
| | | A, ES:[HL+B] | 3 | 2 | 5 | A ← A∨((ES:HL)+B) | × |
| | | A, [HL+C] | 2 | 1 | 4 | A ← A∨(HL+C) | × |
| | | A, ES:[HL+C] | 3 | 2 | 5 | A ← A∨((ES:HL)+C) | × |
| | XOR | A, #byte | 2 | 1 | – | A ← A$\overline{\vee}$byte | × |
| | | saddr, #byte | 3 | 2 | – | (saddr) ← (saddr)$\overline{\vee}$byte | × |
| | | A, r Note 3 | 2 | 1 | – | A ← A$\overline{\vee}$r | × |
| | | r, A | 2 | 1 | – | r ← r$\overline{\vee}$A | × |
| | | A, !addr16 | 3 | 1 | 4 | A ← A$\overline{\vee}$(addr16) | × |
| | | A, ES:!addr16 | 4 | 2 | 5 | A ← A$\overline{\vee}$(ES:addr16) | × |
| | | A, saddr | 2 | 1 | – | A ← A$\overline{\vee}$(saddr) | × |
| | | A, [HL] | 1 | 1 | 4 | A ← A$\overline{\vee}$(HL) | × |
| | | A, ES:[HL] | 2 | 2 | 5 | A ← A$\overline{\vee}$(ES:HL) | × |
| | | A, [HL+byte] | 2 | 1 | 4 | A ← A$\overline{\vee}$(HL+byte) | × |
| | | A, ES:[HL+byte] | 3 | 2 | 5 | A ← A$\overline{\vee}$((ES:HL)+byte) | × |
| | | A, [HL+B] | 2 | 1 | 4 | A ← A$\overline{\vee}$(HL+B) | × |
| | | A, ES:[HL+B] | 3 | 2 | 5 | A ← A$\overline{\vee}$((ES:HL)+B) | × |
| | | A, [HL+C] | 2 | 1 | 4 | A ← A$\overline{\vee}$(HL+C) | × |
| | | A, ES:[HL+C] | 3 | 2 | 5 | A ← A$\overline{\vee}$((ES:HL)+C) | × |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

<R> **2.** Number of CPU clocks ($f_{CLK}$) when the code flash memory is accessed, or when the data flash memory is accessed by an 8-bit instruction.

**3.** Except r = A

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 28-5. Operation List (10/17)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | Flag | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | Note 1 | Note 2 | | Z | AC | CY |
| 8-bit operation | CMP | A, #byte | 2 | 1 | – | A – byte | × | × | × |
| | | !addr16, #byte | 4 | 1 | 4 | (addr16) – byte | × | × | × |
| | | ES:!addr16, #byte | 5 | 2 | 5 | (ES:addr16) – byte | × | × | × |
| | | saddr, #byte | 3 | 1 | – | (saddr) – byte | × | × | × |
| | | A, r Note3 | 2 | 1 | – | A – r | × | × | × |
| | | r, A | 2 | 1 | – | r – A | × | × | × |
| | | A, !addr16 | 3 | 1 | 4 | A – (addr16) | × | × | × |
| | | A, ES:!addr16 | 4 | 2 | 5 | A – (ES:addr16) | × | × | × |
| | | A, saddr | 2 | 1 | – | A – (saddr) | × | × | × |
| | | A, [HL] | 1 | 1 | 4 | A – (HL) | × | × | × |
| | | A, ES:[HL] | 2 | 2 | 5 | A – (ES:HL) | × | × | × |
| | | A, [HL+byte] | 2 | 1 | 4 | A – (HL+byte) | × | × | × |
| | | A, ES:[HL+byte] | 3 | 2 | 5 | A – ((ES:HL)+byte) | × | × | × |
| | | A, [HL+B] | 2 | 1 | 4 | A – (HL+B) | × | × | × |
| | | A, ES:[HL+B] | 3 | 2 | 5 | A – ((ES:HL)+B) | × | × | × |
| | | A, [HL+C] | 2 | 1 | 4 | A – (HL+C) | × | × | × |
| | | A, ES:[HL+C] | 3 | 2 | 5 | A – ((ES:HL)+C) | × | × | × |
| | CMP0 | A | 1 | 1 | – | A – 00H | × | 0 | 0 |
| | | X | 1 | 1 | – | X – 00H | × | 0 | 0 |
| | | B | 1 | 1 | – | B – 00H | × | 0 | 0 |
| | | C | 1 | 1 | – | C – 00H | × | 0 | 0 |
| | | !addr16 | 3 | 1 | 4 | (addr16) – 00H | × | 0 | 0 |
| | | ES:!addr16 | 4 | 2 | 5 | (ES:addr16) – 00H | × | 0 | 0 |
| | | saddr | 2 | 1 | – | (saddr) – 00H | × | 0 | 0 |
| | CMPS | X, [HL+byte] | 3 | 1 | 4 | X – (HL+byte) | × | × | × |
| | | X, ES:[HL+byte] | 4 | 2 | 5 | X – ((ES:HL)+byte) | × | × | × |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

<R> **2.** Number of CPU clocks ($f_{CLK}$) when the code flash memory is accessed, or when the data flash memory is accessed by an 8-bit instruction.

**3.** Except r = A

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 28-5. Operation List (11/17)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | Flag | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | Note 1 | Note 2 | | Z | AC | CY |
| 16-bit operation | ADDW | AX, #word | 3 | 1 | – | AX, CY ← AX+word | × | × | × |
| | | AX, AX | 1 | 1 | – | AX, CY ← AX+AX | × | × | × |
| | | AX, BC | 1 | 1 | – | AX, CY ← AX+BC | × | × | × |
| | | AX, DE | 1 | 1 | – | AX, CY ← AX+DE | × | × | × |
| | | AX, HL | 1 | 1 | – | AX, CY ← AX+HL | × | × | × |
| | | AX, !addr16 | 3 | 1 | 4 | AX, CY ← AX+(addr16) | × | × | × |
| | | AX, ES:!addr16 | 4 | 2 | 5 | AX, CY ← AX+(ES:addr16) | × | × | × |
| | | AX, saddrp | 2 | 1 | – | AX, CY ← AX+(saddrp) | × | × | × |
| | | AX, [HL+byte] | 3 | 1 | 4 | AX, CY ← AX+(HL+byte) | × | × | × |
| | | AX, ES: [HL+byte] | 4 | 2 | 5 | AX, CY ← AX+((ES:HL)+byte) | × | × | × |
| | SUBW | AX, #word | 3 | 1 | – | AX, CY ← AX – word | × | × | × |
| | | AX, BC | 1 | 1 | – | AX, CY ← AX – BC | × | × | × |
| | | AX, DE | 1 | 1 | – | AX, CY ← AX – DE | × | × | × |
| | | AX, HL | 1 | 1 | – | AX, CY ← AX – HL | × | × | × |
| | | AX, !addr16 | 3 | 1 | 4 | AX, CY ← AX – (addr16) | × | × | × |
| | | AX, ES:!addr16 | 4 | 2 | 5 | AX, CY ← AX – (ES:addr16) | × | × | × |
| | | AX, saddrp | 2 | 1 | – | AX, CY ← AX – (saddrp) | × | × | × |
| | | AX, [HL+byte] | 3 | 1 | 4 | AX, CY ← AX – (HL+byte) | × | × | × |
| | | AX, ES: [HL+byte] | 4 | 2 | 5 | AX, CY ← AX – ((ES:HL)+byte) | × | × | × |
| | CMPW | AX, #word | 3 | 1 | – | AX – word | × | × | × |
| | | AX, BC | 1 | 1 | – | AX – BC | × | × | × |
| | | AX, DE | 1 | 1 | – | AX – DE | × | × | × |
| | | AX, HL | 1 | 1 | – | AX – HL | × | × | × |
| | | AX, !addr16 | 3 | 1 | 4 | AX – (addr16) | × | × | × |
| | | AX, ES:!addr16 | 4 | 2 | 5 | AX – (ES:addr16) | × | × | × |
| | | AX, saddrp | 2 | 1 | – | AX – (saddrp) | × | × | × |
| | | AX, [HL+byte] | 3 | 1 | 4 | AX – (HL+byte) | × | × | × |
| | | AX, ES: [HL+byte] | 4 | 2 | 5 | AX – ((ES:HL)+byte) | × | × | × |
| Multiply | MULU | X | 1 | 1 | – | AX ← A×X | | | |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

<R> **2.** Number of CPU clocks ($f_{CLK}$) when the code flash memory is accessed, or when the data flash memory is accessed by an 8-bit instruction.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 28-5. Operation List (12/17)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | Flag | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | Note 1 | Note 2 | | Z | AC | CY |
| Increment/ decrement | INC | r | 1 | 1 | – | r ← r+1 | × | × | |
| | | !addr16 | 3 | 2 | – | (addr16) ← (addr16)+1 | × | × | |
| | | ES:!addr16 | 4 | 3 | – | (ES, addr16) ← (ES, addr16)+1 | × | × | |
| | | saddr | 2 | 2 | – | (saddr) ← (saddr)+1 | × | × | |
| | | [HL+byte] | 3 | 2 | – | (HL+byte) ← (HL+byte)+1 | × | × | |
| | | ES: [HL+byte] | 4 | 3 | – | ((ES:HL)+byte) ← ((ES:HL)+byte)+1 | × | × | |
| | DEC | r | 1 | 1 | – | r ← r – 1 | × | × | |
| | | !addr16 | 3 | 2 | – | (addr16) ← (addr16) – 1 | × | × | |
| | | ES:!addr16 | 4 | 3 | – | (ES, addr16) ← (ES, addr16) – 1 | × | × | |
| | | saddr | 2 | 2 | – | (saddr) ← (saddr) – 1 | × | × | |
| | | [HL+byte] | 3 | 2 | – | (HL+byte) ← (HL+byte) – 1 | × | × | |
| | | ES: [HL+byte] | 4 | 3 | – | ((ES:HL)+byte) ← ((ES:HL)+byte) – 1 | × | × | |
| | INCW | rp | 1 | 1 | – | rp ← rp+1 | | | |
| | | !addr16 | 3 | 2 | – | (addr16) ← (addr16)+1 | | | |
| | | ES:!addr16 | 4 | 3 | – | (ES, addr16) ← (ES, addr16)+1 | | | |
| | | saddrp | 2 | 2 | – | (saddrp) ← (saddrp)+1 | | | |
| | | [HL+byte] | 3 | 2 | – | (HL+byte) ← (HL+byte)+1 | | | |
| | | ES: [HL+byte] | 4 | 3 | – | ((ES:HL)+byte) ← ((ES:HL)+byte)+1 | | | |
| | DECW | rp | 1 | 1 | – | rp ← rp – 1 | | | |
| | | !addr16 | 3 | 2 | – | (addr16) ← (addr16) – 1 | | | |
| | | ES:!addr16 | 4 | 3 | – | (ES, addr16) ← (ES, addr16) – 1 | | | |
| | | saddrp | 2 | 2 | – | (saddrp) ← (saddrp) – 1 | | | |
| | | [HL+byte] | 3 | 2 | – | (HL+byte) ← (HL+byte) – 1 | | | |
| | | ES: [HL+byte] | 4 | 3 | – | ((ES:HL)+byte) ← ((ES:HL)+byte) – 1 | | | |
| Shift | SHR | A, cnt | 2 | 1 | – | (CY ← $A_0$, $A_{m-1}$ ← $A_m$, $A_7$ ← 0) ×cnt | | | × |
| | SHRW | AX, cnt | 2 | 1 | – | (CY ← $AX_0$, $AX_{m-1}$ ← $AX_m$, $AX_{15}$ ← 0) ×cnt | | | × |
| | SHL | A, cnt | 2 | 1 | – | (CY ← $A_7$, $A_m$ ← $A_{m-1}$, $A_0$ ← 0) ×cnt | | | × |
| | | B, cnt | 2 | 1 | – | (CY ← $B_7$, $B_m$ ← $B_{m-1}$, $B_0$ ← 0) ×cnt | | | × |
| | | C, cnt | 2 | 1 | – | (CY ← $C_7$, $C_m$ ← $C_{m-1}$, $C_0$ ← 0) ×cnt | | | × |
| | SHLW | AX, cnt | 2 | 1 | – | (CY ← $AX_{15}$, $AX_m$ ← $AX_{m-1}$, $AX_0$ ← 0) ×cnt | | | × |
| | | BC, cnt | 2 | 1 | – | (CY ← $BC_{15}$, $BC_m$ ← $BC_{m-1}$, $BC_0$ ← 0) ×cnt | | | × |
| | SAR | A, cnt | 2 | 1 | – | (CY ← $A_0$, $A_{m-1}$ ← $A_m$, $A_7$ ← $A_7$) ×cnt | | | × |
| | SARW | AX, cnt | 2 | 1 | – | (CY ← $AX_0$, $AX_{m-1}$ ← $AX_m$, $AX_{15}$← $AX_{15}$) ×cnt | | | × |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

<R> **2.** Number of CPU clocks ($f_{CLK}$) when the code flash memory is accessed, or when the data flash memory is accessed by an 8-bit instruction.

**Remarks 1.** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**2.** cnt indicates the bit shift count.

**Table 28-5. Operation List (13/17)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | Flag | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | Note 1 | Note 2 | | Z | AC | CY |
| Rotate | ROR | A, 1 | 2 | 1 | – | (CY, $A_7 \leftarrow A_0$, $A_{m-1} \leftarrow A_m$)×1 | | | × |
| | ROL | A, 1 | 2 | 1 | – | (CY, $A_0 \leftarrow A_7$, $A_{m+1} \leftarrow A_m$)×1 | | | × |
| | RORC | A, 1 | 2 | 1 | – | (CY $\leftarrow A_0$, $A_7 \leftarrow$ CY, $A_{m-1} \leftarrow A_m$)×1 | | | × |
| | ROLC | A, 1 | 2 | 1 | – | (CY $\leftarrow A_7$, $A_0 \leftarrow$ CY, $A_{m+1} \leftarrow A_m$)×1 | | | × |
| | ROLWC | AX,1 | 2 | 1 | – | (CY $\leftarrow AX_{15}$, $AX_0 \leftarrow$ CY, $AX_{m+1} \leftarrow AX_m$) ×1 | | | × |
| | | BC,1 | 2 | 1 | – | (CY $\leftarrow BC_{15}$, $BC_0 \leftarrow$ CY, $BC_{m+1} \leftarrow BC_m$) ×1 | | | × |
| Bit manipulate | MOV1 | CY, A.bit | 2 | 1 | – | CY ← A.bit | | | × |
| | | A.bit, CY | 2 | 1 | – | A.bit ← CY | | | |
| | | CY, PSW.bit | 3 | 1 | – | CY ← PSW.bit | | | × |
| | | PSW.bit, CY | 3 | 4 | – | PSW.bit ← CY | × | × | |
| | | CY, saddr.bit | 3 | 1 | – | CY ← (saddr).bit | | | × |
| | | saddr.bit, CY | 3 | 2 | – | (saddr).bit ← CY | | | |
| | | CY, sfr.bit | 3 | 1 | – | CY ← sfr.bit | | | × |
| | | sfr.bit, CY | 3 | 2 | – | sfr.bit ← CY | | | |
| | | CY,[HL].bit | 2 | 1 | 4 | CY ← (HL).bit | | | × |
| | | [HL].bit, CY | 2 | 2 | – | (HL).bit ← CY | | | |
| | | CY, ES:[HL].bit | 3 | 2 | 5 | CY ← (ES, HL).bit | | | × |
| | | ES:[HL].bit, CY | 3 | 3 | – | (ES, HL).bit ← CY | | | |
| | AND1 | CY, A.bit | 2 | 1 | – | CY ← CY ∧ A.bit | | | × |
| | | CY, PSW.bit | 3 | 1 | – | CY ← CY ∧ PSW.bit | | | × |
| | | CY, saddr.bit | 3 | 1 | – | CY ← CY ∧ (saddr).bit | | | × |
| | | CY, sfr.bit | 3 | 1 | – | CY ← CY ∧ sfr.bit | | | × |
| | | CY,[HL].bit | 2 | 1 | 4 | CY ← CY ∧ (HL).bit | | | × |
| | | CY, ES:[HL].bit | 3 | 2 | 5 | CY ← CY ∧ (ES, HL).bit | | | × |
| | OR1 | CY, A.bit | 2 | 1 | – | CY ← CY ∨ A.bit | | | × |
| | | CY, PSW.bit | 3 | 1 | – | CYX ← CY ∨ ∨PSW.bit | | | × |
| | | CY, saddr.bit | 3 | 1 | – | CY ← CY ∨ (saddr).bit | | | × |
| | | CY, sfr.bit | 3 | 1 | – | CY ← CY ∨ sfr.bit | | | × |
| | | CY, [HL].bit | 2 | 1 | 4 | CY ← CY ∨ (HL).bit | | | × |
| | | CY, ES:[HL].bit | 3 | 2 | 5 | CY ← CY ∨ (ES, HL).bit | | | × |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

<R> **2.** Number of CPU clocks ($f_{CLK}$) when the code flash memory is accessed, or when the data flash memory is accessed by an 8-bit instruction.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 28-5. Operation List (14/17)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | Flag | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | Note 1 | Note 2 | | Z | AC | CY |
| Bit manipulate | XOR1 | CY, A.bit | 2 | 1 | – | CY ← CY ⊻ A.bit | | | × |
| | | CY, PSW.bit | 3 | 1 | – | CY ← CY ⊻ PSW.bit | | | × |
| | | CY, saddr.bit | 3 | 1 | – | CY ← CY ⊻ (saddr).bit | | | × |
| | | CY, sfr.bit | 3 | 1 | – | CY ← CY ⊻ sfr.bit | | | × |
| | | CY, [HL].bit | 2 | 1 | 4 | CY ← CY ⊻ (HL).bit | | | × |
| | | CY, ES:[HL].bit | 3 | 2 | 5 | CY ← CY ⊻ (ES, HL).bit | | | × |
| | SET1 | A.bit | 2 | 1 | – | A.bit ← 1 | | | |
| | | PSW.bit | 3 | 4 | – | PSW.bit ← 1 | × | × | × |
| | | !addr16.bit | 4 | 2 | – | (addr16).bit ← 1 | | | |
| | | ES:!addr16.bit | 5 | 3 | – | (ES, addr16).bit ← 1 | | | |
| | | saddr.bit | 3 | 2 | – | (saddr).bit ← 1 | | | |
| | | sfr.bit | 3 | 2 | – | sfr.bit ← 1 | | | |
| | | [HL].bit | 2 | 2 | – | (HL).bit ← 1 | | | |
| | | ES:[HL].bit | 3 | 3 | – | (ES, HL).bit ← 1 | | | |
| | CLR1 | A.bit | 2 | 1 | – | A.bit ← 0 | | | |
| | | PSW.bit | 3 | 4 | – | PSW.bit ← 0 | × | × | × |
| | | !addr16.bit | 4 | 2 | – | (addr16).bit ← 0 | | | |
| | | ES:!addr16.bit | 5 | 3 | – | (ES, addr16).bit ← 0 | | | |
| | | saddr.bit | 3 | 2 | – | (saddr.bit) ← 0 | | | |
| | | sfr.bit | 3 | 2 | – | sfr.bit ← 0 | | | |
| | | [HL].bit | 2 | 2 | – | (HL).bit ← 0 | | | |
| | | ES:[HL].bit | 3 | 3 | – | (ES, HL).bit ← 0 | | | |
| | SET1 | CY | 2 | 1 | – | CY ← 1 | | | 1 |
| | CLR1 | CY | 2 | 1 | – | CY ← 0 | | | 0 |
| | NOT1 | CY | 2 | 1 | – | CY ← $\overline{CY}$ | | | × |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed

<R> **2.** Number of CPU clocks ($f_{CLK}$) when the code flash memory is accessed, or when the data flash memory is accessed by an 8-bit instruction.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 28-5. Operation List (15/17)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | Flag |
|---|---|---|---|---|---|---|---|
| | | | | Note 1 | Note 2 | | Z AC CY |
| Call/ return | CALL | rp | 2 | 3 | – | (SP – 2) ← (PC+2)S, (SP – 3) ← (PC+2)H, (SP – 4) ← (PC+2)L, PC ← CS, rp, SP ← SP – 4 | |
| | | $!addr20 | 3 | 3 | – | (SP – 2) ← (PC+3)S, (SP – 3) ← (PC+3)H, (SP – 4) ← (PC+3)L, PC ← PC+3+jdisp16, SP ← SP – 4 | |
| | | !addr16 | 3 | 3 | – | (SP – 2) ← (PC+3)S, (SP – 3) ← (PC+3)H, (SP – 4) ← (PC+3)L, PC ← 0000, addr16, SP ← SP – 4 | |
| | | !!addr20 | 4 | 3 | – | (SP – 2) ← (PC+4)S, (SP – 3) ← (PC+4)H, (SP – 4) ← (PC+4)L, PC ← addr20, SP ← SP – 4 | |
| | CALLT | [addr5] | 2 | 5 | – | (SP – 2) ← (PC+2)S , (SP – 3) ← (PC+2)H, (SP – 4) ← (PC+2)L , PCS ← 0000, PCH ← (0000, addr5+1), PCL ← (0000, addr5), SP ← SP – 4 | |
| | BRK | – | 2 | 5 | – | (SP – 1) ← PSW, (SP – 2) ← (PC+2)S, (SP – 3) ← (PC+2)H, (SP – 4) ← (PC+2)L, PCS ← 0000, PCH ← (0007FH), PCL ← (0007EH), SP ← SP – 4, IE ← 0 | |
| | RET | – | 1 | 6 | – | PCL ← (SP), PCH ← (SP+1), PCS ← (SP+2), SP ← SP+4 | |
| | RETI | – | 2 | 6 | – | PCL ← (SP), PCH ← (SP+1), PCS ← (SP+2), PSW← (SP+3), SP ← SP+4 | R R R |
| | RETB | – | 2 | 6 | – | PCL ← (SP), PCH ← (SP+1), PCS ← (SP+2), PSW ← (SP+3), SP ← SP+4 | R R R |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

<R> **2.** Number of CPU clocks ($f_{CLK}$) when the code flash memory is accessed, or when the data flash memory is accessed by an 8-bit instruction.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 28-5. Operation List (16/17)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | Flag |
|---|---|---|---|---|---|---|---|
| | | | | Note 1 | Note 2 | | Z AC CY |
| Stack manipulate | PUSH | PSW | 2 | 1 | – | (SP – 1) ← PSW, (SP – 2) ← 00H, SP ← SP–2 | |
| | | rp | 1 | 1 | – | (SP – 1) ← $rp_H$, (SP – 2) ← $rp_L$, SP ← SP – 2 | |
| | POP | PSW | 2 | 3 | – | PSW ← (SP+1), SP ← SP + 2 | R R R |
| | | rp | 1 | 1 | – | $rp_L$ ←(SP), $rp_H$ ← (SP+1), SP ← SP + 2 | |
| | MOVW | SP, #word | 4 | 1 | – | SP ← word | |
| | | SP, AX | 2 | 1 | – | SP ← AX | |
| | | AX, SP | 2 | 1 | – | AX ← SP | |
| | | HL, SP | 3 | 1 | – | HL ← SP | |
| | | BC, SP | 3 | 1 | – | BC ← SP | |
| | | DE, SP | 3 | 1 | – | DE ← SP | |
| | ADDW | SP, #byte | 2 | 1 | – | SP ← SP + byte | |
| | SUBW | SP, #byte | 2 | 1 | – | SP ← SP – byte | |
| Unconditional branch | BR | AX | 2 | 3 | – | PC ← CS, AX | |
| | | $addr20 | 2 | 3 | – | PC ← PC + 2 + jdisp8 | |
| | | $!addr20 | 3 | 3 | – | PC ← PC + 3 + jdisp16 | |
| | | !addr16 | 3 | 3 | – | PC ← 0000, addr16 | |
| | | !!addr20 | 4 | 3 | – | PC ← addr20 | |
| Conditional branch | BC | $addr20 | 2 | 2/4[Note3] | – | PC ← PC + 2 + jdisp8 if CY = 1 | |
| | BNC | $addr20 | 2 | 2/4[Note3] | – | PC ← PC + 2 + jdisp8 if CY = 0 | |
| | BZ | $addr20 | 2 | 2/4[Note3] | – | PC ← PC + 2 + jdisp8 if Z = 1 | |
| | BNZ | $addr20 | 2 | 2/4[Note3] | – | PC ← PC + 2 + jdisp8 if Z = 0 | |
| | BH | $addr20 | 3 | 2/4[Note3] | – | PC ← PC + 3 + jdisp8 if (Z∨CY)=0 | |
| | BNH | $addr20 | 3 | 2/4[Note3] | – | PC ← PC + 3 + jdisp8 if (Z∨CY)=1 | |
| | BT | saddr.bit, $addr20 | 4 | 3/5[Note3] | – | PC ← PC + 4 + jdisp8 if (saddr).bit = 1 | |
| | | sfr.bit, $addr20 | 4 | 3/5[Note3] | – | PC ← PC + 4 + jdisp8 if sfr.bit = 1 | |
| | | A.bit, $addr20 | 3 | 3/5[Note3] | – | PC ← PC + 3 + jdisp8 if A.bit = 1 | |
| | | PSW.bit, $addr20 | 4 | 3/5[Note3] | – | PC ← PC + 4 + jdisp8 if PSW.bit = 1 | |
| | | [HL].bit, $addr20 | 3 | 3/5[Note3] | 6/7 | PC ← PC + 3 + jdisp8 if (HL).bit = 1 | |
| | | ES:[HL].bit, $addr20 | 4 | 4/6[Note3] | 7/8 | PC ← PC + 4 + jdisp8 if (ES, HL).bit = 1 | |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

<R> **2.** Number of CPU clocks ($f_{CLK}$) when the code flash memory is accessed, or when the data flash memory is accessed by an 8-bit instruction.

**3.** This indicates the number of clocks “when condition is not met/when condition is met”.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**Table 28-5. Operation List (17/17)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Clocks | Flag |
|---|---|---|---|---|---|---|---|
| | | | | Note 1 | Note 2 | | Z AC CY |
| Conditional branch | BF | saddr.bit, $addr20 | 4 | 3/5[Note 3] | – | PC ← PC + 4 + jdisp8 if (saddr).bit = 0 | |
| | | sfr.bit, $addr20 | 4 | 3/5[Note 3] | – | PC ← PC + 4 + jdisp8 if sfr.bit = 0 | |
| | | A.bit, $addr20 | 3 | 3/5[Note 3] | – | PC ← PC + 3 + jdisp8 if A.bit = 0 | |
| | | PSW.bit, $addr20 | 4 | 3/5[Note 3] | – | PC ← PC + 4 + jdisp8 if PSW.bit = 0 | |
| | | [HL].bit, $addr20 | 3 | 3/5[Note 3] | 6/7 | PC ← PC + 3 + jdisp8 if (HL).bit = 0 | |
| | | ES:[HL].bit, $addr20 | 4 | 4/6[Note 3] | 7/8 | PC ← PC + 4 + jdisp8 if (ES, HL).bit = 0 | |
| | BTCLR | saddr.bit, $addr20 | 4 | 3/5[Note 3] | – | PC ← PC + 4 + jdisp8 if (saddr).bit = 1 then reset (saddr).bit | |
| | | sfr.bit, $addr20 | 4 | 3/5[Note 3] | – | PC ← PC + 4 + jdisp8 if sfr.bit = 1 then reset sfr.bit | |
| | | A.bit, $addr20 | 3 | 3/5[Note 3] | – | PC ← PC + 3 + jdisp8 if A.bit = 1 then reset A.bit | |
| | | PSW.bit, $addr20 | 4 | 3/5[Note 3] | – | PC ← PC + 4 + jdisp8 if PSW.bit = 1 then reset PSW.bit | × × × |
| | | [HL].bit, $addr20 | 3 | 3/5[Note 3] | – | PC ← PC + 3 + jdisp8 if (HL).bit = 1 then reset (HL).bit | |
| | | ES:[HL].bit, $addr20 | 4 | 4/6[Note3] | – | PC ← PC + 4 + jdisp8 if (ES, HL).bit = 1 then reset (ES, HL).bit | |
| Conditional skip | SKC | – | 2 | 1 | – | Next instruction skip if CY = 1 | |
| | SKNC | – | 2 | 1 | – | Next instruction skip if CY = 0 | |
| | SKZ | – | 2 | 1 | – | Next instruction skip if Z = 1 | |
| | SKNZ | – | 2 | 1 | – | Next instruction skip if Z = 0 | |
| | SKH | – | 2 | 1 | – | Next instruction skip if (Z∨CY)=0 | |
| | SKNH | – | 2 | 1 | – | Next instruction skip if (Z∨CY)=1 | |
| CPU control | SEL[Note 4] | RBn | 2 | 1 | – | RBS[1:0] ← n | |
| | NOP | – | 1 | 1 | – | No Operation | |
| | EI | – | 3 | 4 | – | IE ← 1 (Enable Interrupt) | |
| | DI | – | 3 | 4 | – | IE ← 0 (Disable Interrupt) | |
| | HALT | – | 2 | 3 | – | Set HALT Mode | |
| | STOP | – | 2 | 3 | – | Set STOP Mode | |

**Notes 1.** Number of CPU clocks ($f_{CLK}$) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

<R> **2.** Number of CPU clocks ($f_{CLK}$) when the code flash memory is accessed, or when the data flash memory is accessed by an 8-bit instruction.

**3.** This indicates the number of clocks “when condition is not met/when condition is met”.

**4.** n indicates the number of register banks (n = 0 to 3).

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

# CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = −40 to +85°C)

This chapter describes the following electrical specifications.

Target products A: Consumer applications $T_A$ = −40 to +85°C
R5F10E8AALA, R5F10E8CALA, R5F10E8DALA, R5F10E8EALA
R5F10EBAANA, R5F10EBCANA, R5F10EBDANA, R5F10EBEANA
R5F10EGAAFB, R5F10EGCAFB, R5F10EGDAFB, R5F10EGEAFB
R5F10EGAANA, R5F10EGCANA, R5F10EGDANA, R5F10EGEANA
R5F10ELCAFB, R5F10ELDAFB, R5F10ELEAFB
R5F10ELCABG, R5F10ELDABG, R5F10ELEABG

G: Industrial applications When $T_A$ = −40 to +105°C products is used in the range of $T_A$ = −40 to +85°C
R5F10EBAGNA, R5F10EBCGNA, R5F10EBDGNA, R5F10EBEGNA
R5F10EGAGFB, R5F10EGCGFB, R5F10EGDGFB, R5F10EGEGFB
R5F10EGAGNA, R5F10EGCGNA, R5F10EGDGNA, R5F10EGEGNA
R5F10ELCGFB, R5F10ELDGFB, R5F10ELEGFB

**Cautions 1. The RL78 microcontrollers have an on-chip debug function, which is provided for development and evaluation. Do not use the on-chip debug function in products designated for mass production, because the guaranteed number of rewritable times of the flash memory may be exceeded when this function is used, and product reliability therefore cannot be guaranteed. Renesas Electronics is not liable for problems occurring when the on-chip debug function is used.**

**2. With products not provided with an $EV_{DD0}$ or $EV_{SS0}$ pin, replace $EV_{DD0}$ with $V_{DD}$, or replace $EV_{SS0}$ with $V_{SS}$.**

**3. The pins mounted depend on the product. See 2.1 Port Function to 2.2.1 With functions for each product.**

## 29.1 Absolute Maximum Ratings

**Absolute Maximum Ratings ($T_A$ = 25°C) (1/2)**

| Parameter | Symbols | Conditions | Ratings | Unit |
|---|---|---|---|---|
| Supply voltage | $V_{DD}$ | | −0.5 to +6.5 | V |
| | $EV_{DD0}$ | | −0.5 to +6.5 | V |
| | $AV_{DD}$ | | −0.5 to +4.6 | V |
| | $AV_{REFP}$ | | −0.3 to $AV_{DD}$ +0.3 Note 3 | V |
| | $EV_{SS0}$ | | −0.5 to +0.3 | V |
| | $AV_{SS}$ | | −0.5 to +0.3 | V |
| | $AV_{REFM}$ | | −0.3 to $AV_{DD}$ +0.3 Note 3 and $AV_{REFM} \leq AV_{REFP}$ | V |
| REGC pin input voltage | $V_{IREGC}$ | REGC | −0.3 to +2.8 and −0.3 to $V_{DD}$ +0.3 Note 1 | V |
| Input voltage | $V_{I1}$ | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P70 to P77, P120, P140, P141 | −0.3 to $EV_{DD0}$ +0.3 and −0.3 to $V_{DD}$ +0.3 Note 2 | V |
| | $V_{I2}$ | P60 to P63 (N-ch open-drain) | −0.3 to +6.5 | V |
| | $V_{I3}$ | P121 to P124, P137, EXCLK, EXCLKS, $\overline{\text{RESET}}$ | −0.3 to $V_{DD}$ +0.3 Note 2 | V |
| | $V_{I4}$ | P20 to P27, P150 to P154 | −0.3 to $AV_{DD}$ +0.3 Note 2 | V |
| Output voltage | $V_{O1}$ | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P60 to P63, P70 to P77, P120, P130, P140, P141 | −0.3 to $EV_{DD0}$ +0.3 Note 2 | V |
| | $V_{O2}$ | P20 to P27, P150 to P154 | −0.3 to $AV_{DD}$ +0.3 Note 2 | V |
| Analog input voltage | $V_{AI1}$ | ANI16 to ANI30 | −0.3 to $EV_{DD0}$ +0.3 and −0.3 to $AV_{REF(+)}$ +0.3 Notes 2, 4 | V |
| | $V_{AI2}$ | ANI0 to ANI12 | −0.3 to $AV_{DD}$ +0.3 and −0.3 to $AV_{REF(+)}$ +0.3 Notes 2, 4 | V |

**Notes 1.** Connect the REGC pin to $V_{SS}$ via a capacitor (0.47 to 1 μF). This value regulates the absolute maximum rating of the REGC pin. Do not use this pin with voltage applied to it.

**2.** Must be 6.5 V or lower.

**3.** Must be 4.6 V or lower.

**4.** Do not exceed $AV_{REF(+)}$ + 0.3 V in case of A/D conversion target pin.

**Caution Product quality may suffer if the absolute maximum rating is exceeded even momentarily for any parameter. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions that ensure that the absolute maximum ratings are not exceeded.**

**Remarks 1.** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

**2.** $AV_{REF(+)}$: + side reference voltage of the A/D converter.

**3.** $V_{SS}$: Reference voltage

**Absolute Maximum Ratings ($T_A$ = 25°C) (2/2)**

| Parameter | Symbols | Conditions | | Ratings | Unit |
|---|---|---|---|---|---|
| Output current, high | $I_{OH1}$ | Per pin | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P70 to P77, P120, P130, P140, P141 | −40 | mA |
| | | Total of all pins −170 mA | P00 to P04, P40 to P43, P120, P130, P140, P141 | −70 | mA |
| | | | P05, P06, P10 to P16, P30, P31, P50, P51, P70 to P77, | −100 | mA |
| | $I_{OH2}$ | Per pin | P20 to P27, P150 to P154 | −0.1 | mA |
| | | Total of all pins | | −1.3 | mA |
| Output current, low | $I_{OL1}$ | Per pin | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P60 to P63, P70 to P77, P120, P130, P140, P141 | 40 | mA |
| | | Total of all pins 170 mA | P00 to P04, P40 to P43, P120, P130, P140, P141 | 70 | mA |
| | | | P05, P06, P10 to P16, P30, P31, P50, P51, P60 to P63, P70 to P77 | 100 | mA |
| | $I_{OL2}$ | Per pin | P20 to P27, P150 to P154 | 0.4 | mA |
| | | Total of all pins | | 6.4 | mA |
| Operating ambient temperature | $T_A$ | In normal operation mode | | −40 to +85 | °C |
| | | In flash memory programming mode | | | |
| Storage temperature | $T_{stg}$ | | | −65 to +150 | °C |

**Caution Product quality may suffer if the absolute maximum rating is exceeded even momentarily for any parameter. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions that ensure that the absolute maximum ratings are not exceeded.**

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

## 29.2 Oscillator Characteristics

### 29.2.1 X1, XT1 oscillator characteristics

**($T_A$ = −40 to +85°C, 1.6 V ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = 0 V)**

| Parameter | Resonator | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| X1 clock oscillation frequency ($f_X$)Note | Ceramic resonator/crystal resonator | 2.7 V ≤ $V_{DD}$ ≤ 3.6 V | 1.0 | | 20.0 | MHz |
| | | 2.4 V ≤ $V_{DD}$ < 2.7 V | 1.0 | | 16.0 | MHz |
| | | 1.8 V ≤ $V_{DD}$ < 2.4 V | 1.0 | | 8.0 | MHz |
| | | 1.6 V ≤ $V_{DD}$ < 1.8 V | 1.0 | | 4.0 | MHz |
| XT1 clock oscillation frequency ($f_X$)Note | Crystal resonator | | 32 | 32.768 | 35 | kHz |

**Note** Indicates only permissible oscillator frequency ranges. See AC Characteristics for instruction execution time. Request evaluation by the manufacturer of the oscillator circuit mounted on a board to check the oscillator characteristics.

<R> **Caution Since the CPU is started by the high-speed on-chip oscillator clock after a reset release, check the X1 clock oscillation stabilization time using the oscillation stabilization time counter status register (OSTC) by the user. Determine the oscillation stabilization time of the OSTC register and the oscillation stabilization time select register (OSTS) after sufficiently evaluating the oscillation stabilization time with the resonator to be used.**

<R> **Remark** When using the X1 oscillator and XT1 oscillator, see **5.4 System Clock Oscillator**.

### 29.2.2 On-chip oscillator characteristics

**($T_A$ = −40 to +85°C, 1.6 V ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = 0 V)**

| Oscillators | Parameters | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| High-speed on-chip oscillator clock frequencyNotes 1, 2 | $f_{IH}$ | | | 1 | | 32 | MHz |
| High-speed on-chip oscillator clock frequency accuracy | | −20 to +85 °C | 1.8 V ≤ $V_{DD}$ ≤ 3.6 V | −1.0 | | +1.0 | % |
| | | | 1.6 V ≤ $V_{DD}$ < 1.8 V | −5.0 | | +5.0 | % |
| | | −40 to −20 °C | 1.8 V ≤ $V_{DD}$ ≤ 3.6 V | −1.5 | | +1.5 | % |
| | | | 1.6 V ≤ $V_{DD}$ < 1.8 V | −5.5 | | +5.5 | % |
| Low-speed on-chip oscillator clock frequency | $f_{IL}$ | | | | 15 | | kHz |
| Low-speed on-chip oscillator clock frequency accuracy | | | | −15 | | +15 | % |

**Notes 1.** High-speed on-chip oscillator frequency is selected by bits 0 to 3 of option byte (000C2H/010C2H) and bits 0 to 2 of HOCODIV register.

**2.** This indicates the oscillator characteristics only. See AC Characteristics for instruction execution time.

## 29.3 DC Characteristics

### 29.3.1 Pin characteristics

**($T_A$ = −40 to +85°C, 1.6 V ≤ $AV_{DD}$ ≤ $V_{DD}$ ≤ 3.6 V, 1.6 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)** **(1/5)**

| Items | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Output current, high[Note 1] | $I_{OH1}$ | Per pin for P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P70 to P77, P120, P130, P140, P141 | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | −10.0[Note 2] | mA |
| | | Total of P00 to P04, P40 to P43, P120, P130, P140, P141 (When duty ≤ 70%[Note 3]) | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | −10.0 | mA |
| | | | 1.8 V ≤ $EV_{DD0}$ < 2.7 V | | | −5.0 | mA |
| | | | 1.6 V ≤ $EV_{DD0}$ < 1.8 V | | | −2.5 | mA |
| | | Total of P05, P06, P10 to P16, P30, P31, P50, P51, P70 to P77, (When duty ≤ 70%[Note 3]) | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | −19.0 | mA |
| | | | 1.8 V ≤ $EV_{DD0}$ < 2.7 V | | | −10.0 | mA |
| | | | 1.6 V ≤ $EV_{DD0}$ < 1.8 V | | | −5.0 | mA |
| | | Total of all pins (When duty ≤ 70%[Note 3]) | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | −29.0 | mA |
| | $I_{OH2}$ | Per pin for P20 to P27, P150 to P154 | 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V | | | −0.1[Note 2] | mA |
| | | Total of all pins (When duty ≤ 70%[Note 3]) | 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V | | | −1.3 | mA |

**Notes 1.** Value of current at which the device operation is guaranteed even if the current flows from the $EV_{DD0}$, $V_{DD}$ pins to an output pin.

**2.** However, do not exceed the total current value.

**3.** Specification under conditions where the duty factor ≤ 70%.
The output current value that has changed to the duty factor > 70% the duty ratio can be calculated with the following expression (when changing the duty factor from 70% to n%).

- Total output current of pins = ($I_{OH}$ × 0.7)/(n × 0.01)

<Example> Where n = 80% and $I_{OH}$ = −10.0 mA
Total output current of pins = (−10.0 × 0.7)/(80 × 0.01) ≅ −8.7 mA

However, the current that is allowed to flow into one pin does not vary depending on the duty factor. A current higher than the absolute maximum rating must not flow into one pin.

**Caution P00, P02 to P04, P10 to P15, P43, P50, P71, and P74 do not output high level in N-ch open-drain mode.**

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

**($T_A$ = −40 to +85°C, 1.6 V ≤ $AV_{DD}$ ≤ $V_{DD}$ ≤ 3.6 V, 1.6 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)** **(2/5)**

| Items | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Output current, low[Note 1] | $I_{OL1}$ | Per pin for P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P70 to P77, P120, P130, P140, P141 | | | | 20.0[Note 2] | mA |
| | | Per pin for P60 to P63 | | | | 15.0[Note 2] | mA |
| | | Total of P00 to P04, P40 to P43, P120, P130, P140, P141 (When duty ≤ 70%[Note 3]) | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | 15.0 | mA |
| | | | 1.8 V ≤ $EV_{DD0}$ < 2.7 V | | | 9.0 | mA |
| | | | 1.6 V ≤ $EV_{DD0}$ < 1.8 V | | | 4.5 | mA |
| | | Total of P05, P06, P10 to P16, P30, P31, P50, P51, P60 to P63, P70 to P77 (When duty ≤ 70%[Note 3]) | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | 35.0 | mA |
| | | | 1.8 V ≤ $EV_{DD0}$ < 2.7 V | | | 20.0 | mA |
| | | | 1.6 V ≤ $EV_{DD0}$ < 1.8 V | | | 10.0 | mA |
| | | Total of all pins (When duty ≤ 70%[Note 3]) | | | | 50.0 | mA |
| | $I_{OL2}$ | Per pin for P20 to P27, P150 to P154 | | | | 0.4[Note 2] | mA |
| | | Total of all pins (When duty ≤ 70%[Note 3]) | 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V | | | 5.2 | mA |

**Notes 1.** Value of current at which the device operation is guaranteed even if the current flows from an output pin to the $EV_{SS0}$ and $V_{SS}$ pin.

**2.** However, do not exceed the total current value.

**3.** Specification under conditions where the duty factor ≤ 70%.
The output current value that has changed to the duty factor > 70% the duty ratio can can be calculated with the following expression (when changing the duty factor from 70% to n%).

- Total output current of pins = ($I_{OL}$ × 0.7)/(n × 0.01)

<Example> Where n = 80% and $I_{OL}$ = 10.0 mA
Total output current of pins = (10.0 × 0.7)/(80 × 0.01) ≅ 8.7 mA

However, the current that is allowed to flow into one pin does not vary depending on the duty factor. A current higher than the absolute maximum rating must not flow into one pin.

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

**($T_A$ = −40 to +85°C, 1.6 V ≤ $AV_{DD}$ ≤ $V_{DD}$ ≤ 3.6 V, 1.6 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)** **(3/5)**

| Items | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Input voltage, high | $V_{IH1}$ | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P70 to P77, P120, P140, P141 | Normal input buffer | 0.8$EV_{DD0}$ | | $EV_{DD0}$ | V |
| | $V_{IH2}$ | P01, P03, P04, P10, P11, P13 to P16, P43 | TTL input buffer 3.3 V ≤ $EV_{DD0}$ ≤ 3.6 V | 2.0 | | $EV_{DD0}$ | V |
| | | | TTL input buffer 1.6 V ≤ $EV_{DD0}$ < 3.3 V | 1.5 | | $EV_{DD0}$ | V |
| | $V_{IH3}$ | P20 to P27, P150 to P154 | | 0.7$AV_{DD}$ | | $AV_{DD}$ | V |
| | $V_{IH4}$ | P60 to P63 | | 0.7$EV_{DD0}$ | | 6.0 | V |
| | $V_{IH5}$ | P121 to P124, P137, EXCLK, EXCLKS, $\overline{\text{RESET}}$ | | 0.8$V_{DD}$ | | $V_{DD}$ | V |
| Input voltage, low | $V_{IL1}$ | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P70 to P77, P120, P140, P141 | Normal input buffer | 0 | | 0.2$EV_{DD0}$ | V |
| | $V_{IL2}$ | P01, P03, P04, P10, P11, P13 to P16, P43 | TTL input buffer 3.3 V ≤ $EV_{DD0}$ ≤ 3.6 V | 0 | | 0.5 | V |
| | | | TTL input buffer 1.6 V ≤ $EV_{DD0}$ < 3.3 V | 0 | | 0.32 | V |
| | $V_{IL3}$ | P20 to P27, P150 to P154 | | 0 | | 0.3$AV_{DD}$ | V |
| | $V_{IL4}$ | P60 to P63 | | 0 | | 0.3$EV_{DD0}$ | V |
| | $V_{IL5}$ | P121 to P124, P137, EXCLK, EXCLKS, $\overline{\text{RESET}}$ | | 0 | | 0.2$V_{DD}$ | V |

**Caution The maximum value of $V_{IH}$ of pins P00, P02 to P04, P10 to P15, P43, P50, P71, and P74 is $EV_{DD0}$, even in the N-ch open-drain mode.**

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

**($T_A$ = −40 to +85°C, 1.6 V ≤ $AV_{DD}$ ≤ $V_{DD}$ ≤ 3.6 V, 1.6 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)** **(4/5)**

| Items | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Output voltage, high | $V_{OH1}$ | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P70 to P77, P120, P130, P140, P141 | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, $I_{OH1}$ = −2.0 mA | $EV_{DD0}$ − 0.6 | | | V |
| | | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V, $I_{OH1}$ = −1.5 mA | $EV_{DD0}$ − 0.5 | | | V |
| | | | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V, $I_{OH1}$ = −1.0 mA | $EV_{DD0}$ − 0.5 | | | V |
| | $V_{OH2}$ | P20 to P27, P150 to P154 | 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V, $I_{OH2}$ = −100 μA | $AV_{DD}$ − 0.5 | | | V |
| Output voltage, low | $V_{OL1}$ | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P70 to P77, P120, P130, P140, P141 | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, $I_{OL1}$ = 3.0 mA | | | 0.6 | V |
| | | | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, $I_{OL1}$ = 1.5 mA | | | 0.4 | V |
| | | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V, $I_{OL1}$ = 0.6 mA | | | 0.4 | V |
| | | | 1.6 V ≤ $EV_{DD0}$ < 1.8 V, $I_{OL1}$ = 0.3 mA | | | 0.4 | V |
| | $V_{OL2}$ | P20 to P27, P150 to P154 | 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V, $I_{OL2}$ = 400 μA | | | 0.4 | V |
| | $V_{OL3}$ | P60 to P63 | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, $I_{OL3}$ = 3.0 mA | | | 0.4 | V |
| | | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V, $I_{OL3}$ = 2.0 mA | | | 0.4 | V |
| | | | 1.6 V ≤ $EV_{DD0}$ < 1.8 V, $I_{OL3}$ = 1.0 mA | | | 0.4 | V |

**Caution P00, P02 to P04, P10 to P15, P43, P50, P71, and P74 do not output high level in N-ch open-drain mode.**

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

**($T_A$ = −40 to +85°C, 1.6 V ≤ $AV_{DD}$ ≤ $V_{DD}$ ≤ 3.6 V, 1.6 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)** **(5/5)**

| Items | Symbol | Conditions | | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|---|
| Input leakage current, high | $I_{LIH1}$ | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P60 to P63, P70 to P77, P120, P140, P141 | $V_I$ = $EV_{DD0}$ | | | | 1 | μA |
| | $I_{LIH2}$ | P137, $\overline{\text{RESET}}$ | $V_I$ = $V_{DD}$ | | | | 1 | μA |
| | $I_{LIH3}$ | P121 to P124 (X1, X2, XT1, XT2, EXCLK, EXCLKS) | $V_I$ = $V_{DD}$ | In input port or external clock input | | | 1 | μA |
| | | | | In resonator connection | | | 10 | μA |
| | $I_{LIH4}$ | P20 to P27, P150 to P154 | $V_I$ = $AV_{DD}$ | | | | 1 | μA |
| Input leakage current, low | $I_{LIL1}$ | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P60 to P67, P70 to P77, P120, P140, P141 | $V_I$ = $EV_{SS0}$ | | | | −1 | μA |
| | $I_{LIL2}$ | P137, $\overline{\text{RESET}}$ | $V_I$ = $V_{SS}$ | | | | −1 | μA |
| | $I_{LIL3}$ | P121 to P124 (X1, X2, XT1, XT2, EXCLK, EXCLKS) | $V_I$ = $V_{SS}$ | In input port or external clock input | | | −1 | μA |
| | | | | In resonator connection | | | −10 | μA |
| | $I_{LIL4}$ | P20 to P27, P150 to P154 | $V_I$ = $AV_{SS}$ | | | | −1 | μA |
| On-chip pull-up resistance | $R_U$ | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P70 to P77, P120, P140, P141 | $V_I$ = $EV_{SS0}$, In input port | | 10 | 20 | 100 | kΩ |

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

### 29.3.2 Supply current characteristics

**($T_A$ = −40 to +85°C, 1.6 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)** **(1/3)**

| Parameter | Symbol | Conditions | | | | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| Supply current[Note 1] | $I_{DD1}$ | Operating mode | HS (high-speed main) mode[Note 5] | $f_{IH}$ = 32 MHz[Note 3] | Basic operation | $V_{DD}$ = 3.0 V | | 2.1 | | mA |
| | | | | | Normal operation | $V_{DD}$ = 3.0 V | | 4.6 | 7.0 | mA |
| | | | | $f_{IH}$ = 24 MHz[Note 3] | Normal operation | $V_{DD}$ = 3.0 V | | 3.7 | 5.5 | mA |
| | | | | $f_{IH}$ = 16 MHz[Note 3] | Normal operation | $V_{DD}$ = 3.0 V | | 2.7 | 4.0 | mA |
| | | | LS (low-speed main) mode[Note 5] | $f_{IH}$ = 8 MHz[Note 3] | Normal operation | $V_{DD}$ = 3.0 V | | 1.2 | 1.8 | mA |
| | | | | | | $V_{DD}$ = 2.0 V | | 1.2 | 1.8 | |
| | | | LV (Low-voltage main) mode[Note 5] | $f_{IH}$ = 4 MHz[Note 3] | Normal operation | $V_{DD}$ = 3.0 V | | 1.2 | 1.7 | mA |
| | | | | | | $V_{DD}$ = 2.0 V | | 1.2 | 1.7 | |
| | | | HS (high-speed main) mode[Note 5] | $f_{MX}$ = 20 MHz[Note 2], $V_{DD}$ = 3.0 V | Normal operation | Square wave input | | 3.0 | 4.6 | mA |
| | | | | | | Resonator connection | | 3.2 | 4.8 | |
| | | | | $f_{MX}$ = 10 MHz[Note 2], $V_{DD}$ = 3.0 V | Normal operation | Square wave input | | 1.9 | 2.7 | mA |
| | | | | | | Resonator connection | | 1.9 | 2.7 | |
| | | | LS (low-speed main) mode[Note 5] | $f_{MX}$ = 8 MHz[Note 2], $V_{DD}$ = 3.0 V | Normal operation | Square wave input | | 1.1 | 1.7 | mA |
| | | | | | | Resonator connection | | 1.1 | 1.7 | |
| | | | | $f_{MX}$ = 8 MHz[Note 2], $V_{DD}$ = 2.0 V | Normal operation | Square wave input | | 1.1 | 1.7 | mA |
| | | | | | | Resonator connection | | 1.1 | 1.7 | |
| | | | Subsystem clock mode | $f_{SUB}$ = 32.768 kHz[Note 4] $T_A$ = −40°C | Normal operation | Square wave input | | 4.1 | 4.9 | μA |
| | | | | | | Resonator connection | | 4.2 | 5.0 | |
| | | | | $f_{SUB}$ = 32.768 kHz[Note 4] $T_A$ = +25°C | Normal operation | Square wave input | | 4.2 | 4.9 | μA |
| | | | | | | Resonator connection | | 4.3 | 5.0 | |
| | | | | $f_{SUB}$ = 32.768 kHz[Note 4] $T_A$ = +50°C | Normal operation | Square wave input | | 4.3 | 5.5 | μA |
| | | | | | | Resonator connection | | 4.4 | 5.6 | |
| | | | | $f_{SUB}$ = 32.768 kHz[Note 4] $T_A$ = +70°C | Normal operation | Square wave input | | 4.5 | 6.3 | μA |
| | | | | | | Resonator connection | | 4.6 | 6.4 | |
| | | | | $f_{SUB}$ = 32.768 kHz[Note 4] $T_A$ = +85°C | Normal operation | Square wave input | | 4.8 | 7.7 | μA |
| | | | | | | Resonator connection | | 4.9 | 7.8 | |

(**Notes** and **Remarks** are listed on the next page.)

**Notes 1.** Total current flowing into $V_{DD}$ and $EV_{DD0}$, including the input leakage current flowing when the level of the input pin is fixed to $V_{DD}$, $EV_{DD0}$ or $V_{SS}$, $EV_{SS0}$. The values below the MAX. column include the peripheral operation current. However, not including the current flowing into the A/D converter, LVD circuit, I/O port, on-chip pull-up/pull-down resistors, and data flash rewriting.

**2.** When high-speed on-chip oscillator and subsystem clock are stopped.

**3.** When high-speed system clock and subsystem clock are stopped.

**4.** When high-speed on-chip oscillator and high-speed system clock are stopped. When setting ultra-low power consumption oscillation (AMPHS1 = 1). Not including the current flowing into the RTC, 12-bit interval timer and watchdog timer

**5.** Relationship between operation voltage width, operation frequency of CPU and operation mode is as below.

HS (high-speed main) mode: $V_{DD}$ = 2.7 V to 3.6 V@1 MHz to 32 MHz
$V_{DD}$ = 2.4 V to 3.6 V@1 MHz to 16 MHz
LS (low-speed main) mode: $V_{DD}$ = 1.8 V to 3.6 V@1 MHz to 8 MHz
LV (Low-voltage main) mode: $V_{DD}$ = 1.6 V to 3.6 V@1 MHz to 4 MHz

**Remarks 1.** $f_{MX}$: High-speed system clock frequency (X1 clock oscillation frequency or external main system clock frequency)

**2.** $f_{IH}$: High-speed on-chip oscillator clock frequency

**3.** $f_{SUB}$: Subsystem clock frequency (XT1 clock oscillation frequency)

**4.** Except subsystem clock operation, temperature condition of the TYP. value is $T_A$ = 25°C

**($T_A$ = −40 to +85°C, 1.6 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)** **(2/3)**

| Parameter | Symbol | Conditions | | | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|---|---|
| Supply current[Note 1] | $I_{DD2}$[Note 2] | HALT mode | HS (high-speed main) mode[Note 7] | $f_{IH}$ = 32 MHz[Note 4] | $V_{DD}$ = 3.0 V | | 0.54 | 1.63 | mA |
| | | | | $f_{IH}$ = 24 MHz[Note 4] | $V_{DD}$ = 3.0 V | | 0.44 | 1.28 | mA |
| | | | | $f_{IH}$ = 16 MHz[Note 4] | $V_{DD}$ = 3.0 V | | 0.40 | 1.00 | mA |
| | | | LS (low-speed main) mode[Note 7] | $f_{IH}$ = 8 MHz[Note 4] | $V_{DD}$ = 3.0 V | | 270 | 530 | μA |
| | | | | | $V_{DD}$ = 2.0 V | | 270 | 530 | |
| | | | LV (Low-voltage main) mode[Note 7] | $f_{IH}$ = 4 MHz[Note 4] | $V_{DD}$ = 3.0 V | | 435 | 640 | μA |
| | | | | | $V_{DD}$ = 2.0 V | | 435 | 640 | |
| | | | HS (high-speed main) mode[Note 7] | $f_{MX}$ = 20 MHz[Note 3], $V_{DD}$ = 3.0 V | Square wave input | | 0.28 | 1.00 | mA |
| | | | | | Resonator connection | | 0.45 | 1.17 | |
| | | | | $f_{MX}$ = 10 MHz[Note 3], $V_{DD}$ = 3.0 V | Square wave input | | 0.19 | 0.60 | mA |
| | | | | | Resonator connection | | 0.26 | 0.67 | |
| | | | LS (low-speed main) mode[Note 7] | $f_{MX}$ = 8 MHz[Note 3], $V_{DD}$ = 3.0 V | Square wave input | | 95 | 330 | μA |
| | | | | | Resonator connection | | 145 | 380 | |
| | | | | $f_{MX}$ = 8 MHz[Note 3], $V_{DD}$ = 2.0 V | Square wave input | | 95 | 330 | μA |
| | | | | | Resonator connection | | 145 | 380 | |
| | | | Subsystem clock mode | $f_{SUB}$ = 32.768 kHz[Note 5] $T_A$ = −40°C | Square wave input | | 0.25 | 0.57 | μA |
| | | | | | Resonator connection | | 0.44 | 0.76 | |
| | | | | $f_{SUB}$ = 32.768 kHz[Note 5] $T_A$ = +25°C | Square wave input | | 0.30 | 0.57 | μA |
| | | | | | Resonator connection | | 0.49 | 0.76 | |
| | | | | $f_{SUB}$ = 32.768 kHz[Note 5] $T_A$ = +50°C | Square wave input | | 0.38 | 1.17 | μA |
| | | | | | Resonator connection | | 0.57 | 1.36 | |
| | | | | $f_{SUB}$ = 32.768 kHz[Note 5] $T_A$ = +70°C | Square wave input | | 0.52 | 1.97 | μA |
| | | | | | Resonator connection | | 0.71 | 2.16 | |
| | | | | $f_{SUB}$ = 32.768 kHz[Note 5] $T_A$ = +85°C | Square wave input | | 0.97 | 3.37 | μA |
| | | | | | Resonator connection | | 1.16 | 3.56 | |
| | $I_{DD3}$[Note 6] | STOP mode[Note 8] | $T_A$ = −40°C | | | | 0.16 | 0.50 | μA |
| | | | $T_A$ = +25°C | | | | 0.23 | 0.50 | |
| | | | $T_A$ = +50°C | | | | 0.34 | 1.10 | |
| | | | $T_A$ = +70°C | | | | 0.46 | 1.90 | |
| | | | $T_A$ = +85°C | | | | 0.75 | 3.30 | |

(**Notes** and **Remarks** are listed on the next page.)

**Notes 1.** Total current flowing into $V_{DD}$ and $EV_{DD0}$, including the input leakage current flowing when the level of the input pin is fixed to $V_{DD}$, $EV_{DD0}$ or $V_{SS}$, $EV_{SS0}$. The values below the MAX. column include the peripheral operation current. However, not including the current flowing into the A/D converter, LVD circuit, I/O port, on-chip pull-up/pull-down resistors, and data flash rewriting.

**2.** During HALT instruction execution by flash memory.

**3.** When high-speed on-chip oscillator and subsystem clock are stopped.

**4.** When high-speed system clock and subsystem clock are stopped.

**5.** When high-speed on-chip oscillator and high-speed system clock are stopped. When RTCLPC = 1 and setting ultra-low current consumption (AMPHS1 = 1). Including the current flowing into the RTC. However, not including the current flowing into the 12-bit interval timer, and watchdog timer.

**6.** When subsystem clock is stopped. Not including the current flowing into the RTC, 12-bit interval timer, watchdog timer.

**7.** Relationship between operation voltage width, operation frequency of CPU and operation mode is as below.

HS (high-speed main) mode: $2.7\ V \le V_{DD} \le 3.6\ V$@1 MHz to 32 MHz
$2.4\ V \le V_{DD} \le 3.6\ V$@1 MHz to 16 MHz
LS (low-speed main) mode: $1.8\ V \le V_{DD} < 3.6\ V$@1 MHz to 8 MHz
LV (low-voltage main) mode: $1.6\ V \le V_{DD} \le 3.6\ V$@1 MHz to 4 MHz

**8.** Regarding the value for current to operate the subsystem clock in STOP mode, refer to that in HALT mode.

**Remarks 1.** $f_{MX}$: High-speed system clock frequency (X1 clock oscillation frequency or external main system clock frequency)

**2.** $f_{IH}$: High-speed on-chip oscillator clock frequency

**3.** $f_{SUB}$: Subsystem clock frequency (XT1 clock oscillation frequency)

**4.** Except subsystem clock operation and STOP mode, temperature condition of the TYP. value is $T_A$ = 25°C

<R> **($T_A$ = −40 to +85°C, 1.6 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)** **(3/3)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Low-speed on-chip oscillator operating current | $I_{FIL}$ Note 1 | | | | 0.20 | | μA |
| RTC operating current | $I_{RTC}$ Notes 1, 2, 3 | | | | 0.02 | | μA |
| 12-bit interval timer operating current | $I_{IT}$ Notes 1, 2, 4 | | | | 0.02 | | μA |
| Watchdog timer operating current | $I_{WDT}$ Notes 1, 2, 5 | $f_{IL}$ = 15 kHz | | | 0.22 | | μA |
| A/D converter operating current | $I_{ADC}$ Notes 6, 7 | $AV_{DD}$ = 3.0 V, When conversion at maximum speed | | | 420 | 720 | μA |
| $AV_{REF(+)}$ current | $I_{AVREF}$ Note 8 | $AV_{DD}$ = 3.0 V, ADREFP1 = 0, ADREFP0 = 0 Note 7 | | | 14.0 | 25.0 | μA |
| | | $AV_{REFP}$ = 3.0 V, ADREFP1 = 0, ADREFP0 = 1 Note 10 | | | 14.0 | 25.0 | μA |
| | | ADREFP1 = 1, ADREFP0 = 0 Note 1 | | | 14.0 | 25.0 | μA |
| A/D converter reference voltage current | $I_{ADREF}$ Notes 1, 9 | $V_{DD}$ = 3.0 V | | | 75.0 | | μA |
| Temperature sensor operating current | $I_{TMP}$ Note 1 | $V_{DD}$ = 3.0 V | | | 75.0 | | μA |
| LVD operating current | $I_{LVD}$ Notes 1, 11 | | | | 0.08 | | μA |
| BGO operating current | $I_{BGO}$ Notes 1, 12 | | | | 2.5 | 12.2 | mA |
| Self-programming operating current | $I_{FSP}$ Notes 1, 13 | | | | 2.5 | 12.2 | mA |
| SNOOZE operating current | $I_{SNOZ}$ | A/D converter operation ($AV_{DD}$ = 3.0 V) | The mode is performed Notes 1, 14 | | 0.50 | 0.60 | mA |
| | | | During A/D conversion Note 1 | | 0.60 | 0.75 | mA |
| | | | During A/D conversion Note 7 | | 420 | 720 | μA |
| | | CSI/UART operation Note 1 | | | 0.70 | 0.84 | mA |

(**Notes** and **Remarks** are listed on the next page.)

<R> **Notes 1.** Current flowing to $V_{DD}$.

**2.** When high-speed on-chip oscillator and high-speed system clock are stopped.

**3.** Current flowing only to the real-time clock (RTC) (excluding the operating current of the low-speed on-chip ocsillator and the XT1 oscillator). The supply current of the RL78 microcontrollers is the sum of the values of either $I_{DD1}$ or $I_{DD2}$, and $I_{RTC}$, when the real-time clock operates in operation mode or HALT mode. When the low-speed on-chip oscillator is selected, $I_{FIL}$ should be added. $I_{DD2}$ subsystem clock operation includes the operational current of the real-time clock.

**4.** Current flowing only to the 12-bit interval timer (excluding the operating current of the low-speed on-chip ocsillator and the XT1 oscillator). The supply current of the RL78 microcontrollers is the sum of the values of either $I_{DD1}$ or $I_{DD2}$, and $I_{IT}$, when the 12-bit interval timer operates in operation mode or HALT mode. When the low-speed on-chip oscillator is selected, $I_{FIL}$ should be added.

**5.** Current flowing only to the watchdog timer (including the operating current of the low-speed on-chip oscillator). The supply current of the RL78 microcontrollers is the sum of $I_{DD1}$, $I_{DD2}$, or $I_{DD3}$ and $I_{WDT}$ when the watchdog timer is in operation.

**6.** Current flowing only to the A/D converter. The supply current of the RL78 microcontrollers is the sum of $I_{DD1}$ or $I_{DD2}$ and $I_{ADC}$, $I_{AVREF}$, $I_{ADREF}$ when the A/D converter operates in an operation mode or the HALT mode.

**7.** Current flowing to the $AV_{DD}$.

**8.** Current flowing from the reference voltage source of A/D converter.

**9.** Operation current flowing to the internal reference voltage.

**10.** Current flowing to the $AV_{REFP}$.

**11.** Current flowing only to the LVD circuit. The supply current of the RL78 microcontrollers is the sum of $I_{DD1}$, $I_{DD2}$ or $I_{DD3}$ and $I_{LVD}$ when the LVD circuit is in operation.

**12.** Current flowing only during data flash rewrite.

**13.** Current flowing only during self programming.

**14.** For shift time to the SNOOZE mode, see **18.3.3 SNOOZE mode.**

**Remarks 1.** $f_{IL}$: Low-speed on-chip oscillator clock frequency

**2.** $f_{SUB}$: Subsystem clock frequency (XT1 clock oscillation frequency)

**3.** $f_{CLK}$: CPU/peripheral hardware clock frequency

**4.** Temperature condition of the TYP. value is $T_A$ = 25°C

## 29.4 AC Characteristics

**($T_A$ = −40 to +85°C, $AV_{DD} \le V_{DD} \le 3.6$ V, 1.6 V ≤ $EV_{DD0} \le V_{DD} \le 3.6$ V, $V_{SS} = EV_{SS0}$ = 0 V)**

| Items | Symbol | Conditions | | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|---|
| Instruction cycle (minimum instruction execution time) | $T_{CY}$ | Main system clock ($f_{MAIN}$) operation | HS (high-speed main) mode | 2.7 V ≤ $V_{DD}$ ≤ 3.6 V | 0.03125 | | 1 | μs |
| | | | | 2.4 V ≤ $V_{DD}$ < 2.7 V | 0.0625 | | 1 | μs |
| | | | LS (low-speed main) mode | 1.8 V ≤ $V_{DD}$ ≤ 3.6 V | 0.125 | | 1 | μs |
| | | | LV (low-voltage main) mode | 1.6 V ≤ $V_{DD}$ ≤ 3.6 V | 0.25 | | 1 | μs |
| | | Subsystem clock ($f_{SUB}$) operation | | 1.8 V ≤ $V_{DD}$ ≤ 3.6 V | 28.5 | 30.5 | 31.3 | μs |
| | | In the self programming mode | HS (high-speed main) mode | 2.7 V ≤ $V_{DD}$ ≤ 3.6 V | 0.03125 | | 1 | μs |
| | | | | 2.4 V ≤ $V_{DD}$ < 2.7 V | 0.0625 | | 1 | μs |
| | | | LS (low-speed main) mode | 1.8 V ≤ $V_{DD}$ ≤ 3.6 V | 0.125 | | 1 | μs |
| | | | LV (low-voltage main) mode | 1.6 V ≤ $V_{DD}$ ≤ 3.6 V | 0.25 | | 1 | μs |
| External system clock frequency | $f_{EX}$ | 2.7 V ≤ $V_{DD}$ ≤ 3.6 V | | | 1.0 | | 20.0 | MHz |
| | | 2.4 V ≤ $V_{DD}$ < 2.7 V | | | 1.0 | | 16.0 | MHz |
| | | 1.8 V ≤ $V_{DD}$ < 2.4 V | | | 1.0 | | 8.0 | MHz |
| | | 1.6 V ≤ $V_{DD}$ < 1.8 V | | | 1.0 | | 4.0 | MHz |
| | $f_{EXS}$ | | | | 32 | | 35 | kHz |
| External system clock input high-level width, low-level width | $t_{EXH}$, $t_{EXL}$ | 2.7 V ≤ $V_{DD}$ ≤ 3.6 V | | | 24 | | | ns |
| | | 2.4 V ≤ $V_{DD}$ < 2.7 V | | | 30 | | | ns |
| | | 1.8 V ≤ $V_{DD}$ < 2.4 V | | | 60 | | | ns |
| | | 1.6 V ≤ $V_{DD}$ < 1.8 V | | | 120 | | | ns |
| | $t_{EXHS}$, $t_{EXLS}$ | | | | 13.7 | | | μs |
| TI00, TI01, TI03 to TI07 input high-level width, low-level width | $t_{TIH}$, $t_{TIL}$ | | | | 1/$f_{MCK}$+10 | | | ns[Note] |
| TO00, TO01, TO03 to TO07 output frequency | $f_{TO}$ | HS (high-speed main) mode | | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | 8 | MHz |
| | | | | 1.8 V ≤ $EV_{DD0}$ < 2.7 V | | | 4 | MHz |
| | | | | 1.6 V ≤ $EV_{DD0}$ < 1.8 V | | | 2 | MHz |
| | | LS (low-speed main) mode | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | 4 | MHz |
| | | | | 1.6 V ≤ $EV_{DD0}$ < 1.8 V | | | 2 | MHz |
| | | LV (low-voltage main) mode | | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | 2 | MHz |
| PCLBUZ0, PCLBUZ1 output frequency | $f_{PCL}$ | HS (high-speed main) mode | | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | 8 | MHz |
| | | | | 1.8 V ≤ $EV_{DD0}$ < 2.7 V | | | 4 | MHz |
| | | | | 1.6 V ≤ $EV_{DD0}$ < 1.8 V | | | 2 | MHz |
| | | LS (low-speed main) mode | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | 4 | MHz |
| | | | | 1.6 V ≤ $EV_{DD0}$ < 1.8 V | | | 2 | MHz |
| | | LV (low-voltage main) mode | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | 4 | MHz |
| | | | | 1.6 V ≤ $EV_{DD0}$ < 1.8 V | | | 2 | MHz |
| Interrupt input high-level width, low-level width | $t_{INTH}$, $t_{INTL}$ | INTP0 | | 1.6 V ≤ $V_{DD}$ ≤ 3.6 V | 1 | | | μs |
| | | INTP1 to INTP11 | | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V | 1 | | | μs |
| Key interrupt input high-level width, low-level width | $t_{KR}$ | KR0 to KR9 | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V, 1.8 V ≤ $AV_{DD0}$ ≤ 3.6 V | 250 | | | ns |
| | | | | 1.6 V ≤ $EV_{DD0}$ < 1.8 V, 1.6 V ≤ $AV_{DD0}$ < 1.8 V | 1 | | | μs |
| $\overline{\text{RESET}}$ low-level width | $t_{RSL}$ | | | | 10 | | | μs |

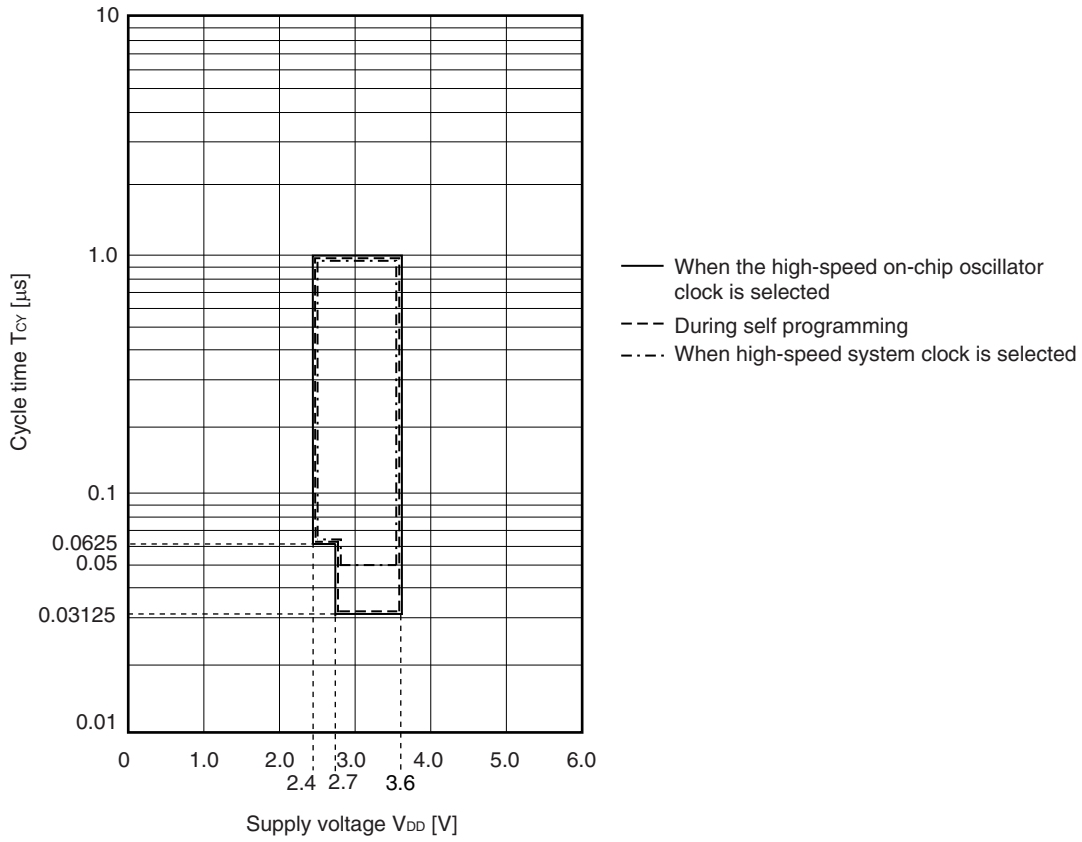
(**Note** and **Remark** are listed on the next page.)

**Note** The following conditions are required for low-voltage interface when $EV_{DD0} < V_{DD}$.
1.8 V ≤ $EV_{DD0}$ < 2.7 V : MIN. 125 ns
1.6 V ≤ $EV_{DD0}$ < 1.8 V : MIN. 250 ns

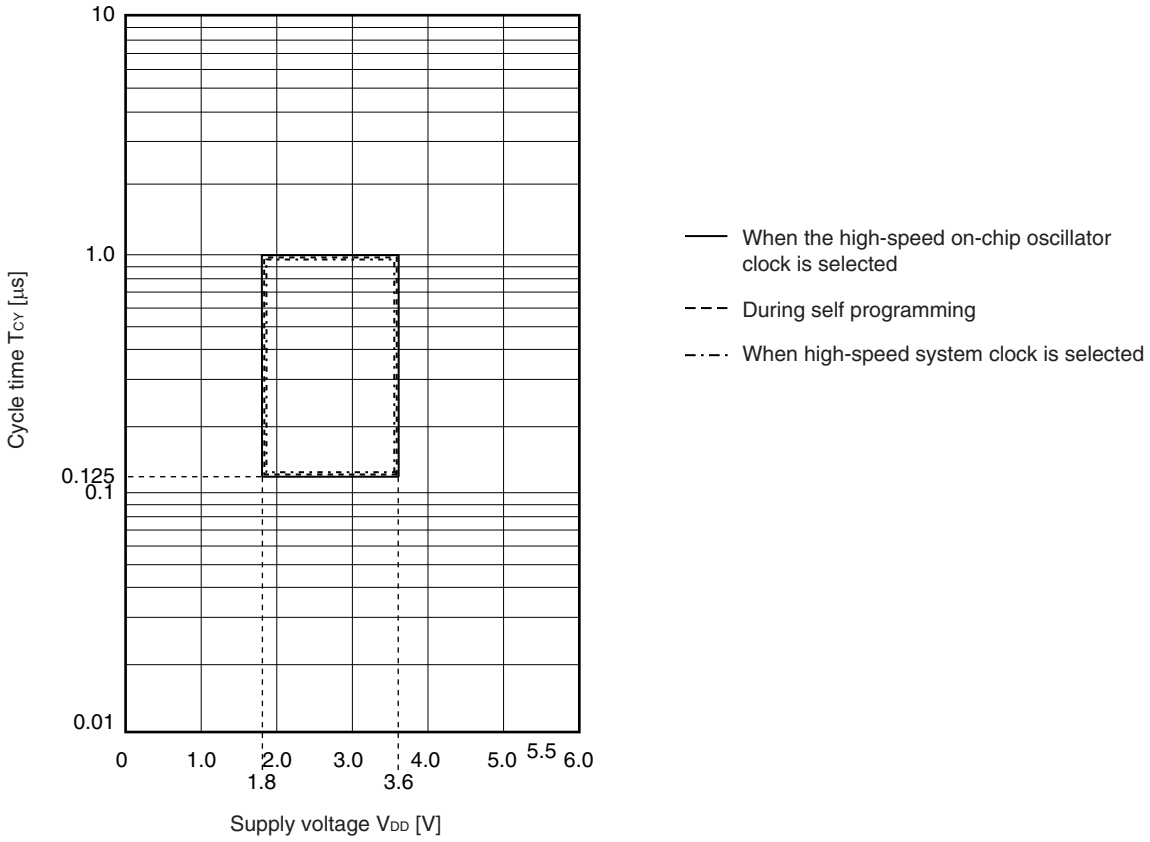
**Remark** $f_{MCK}$: Timer array unit operation clock frequency
(Operation clock to be set by the CKS0n bit of timer clock select register 0 (TPS0) and timer mode register 0n (TMR0n). n: Channel number (n = 0 to 7))

**Minimum Instruction Execution Time during Main System Clock Operation**

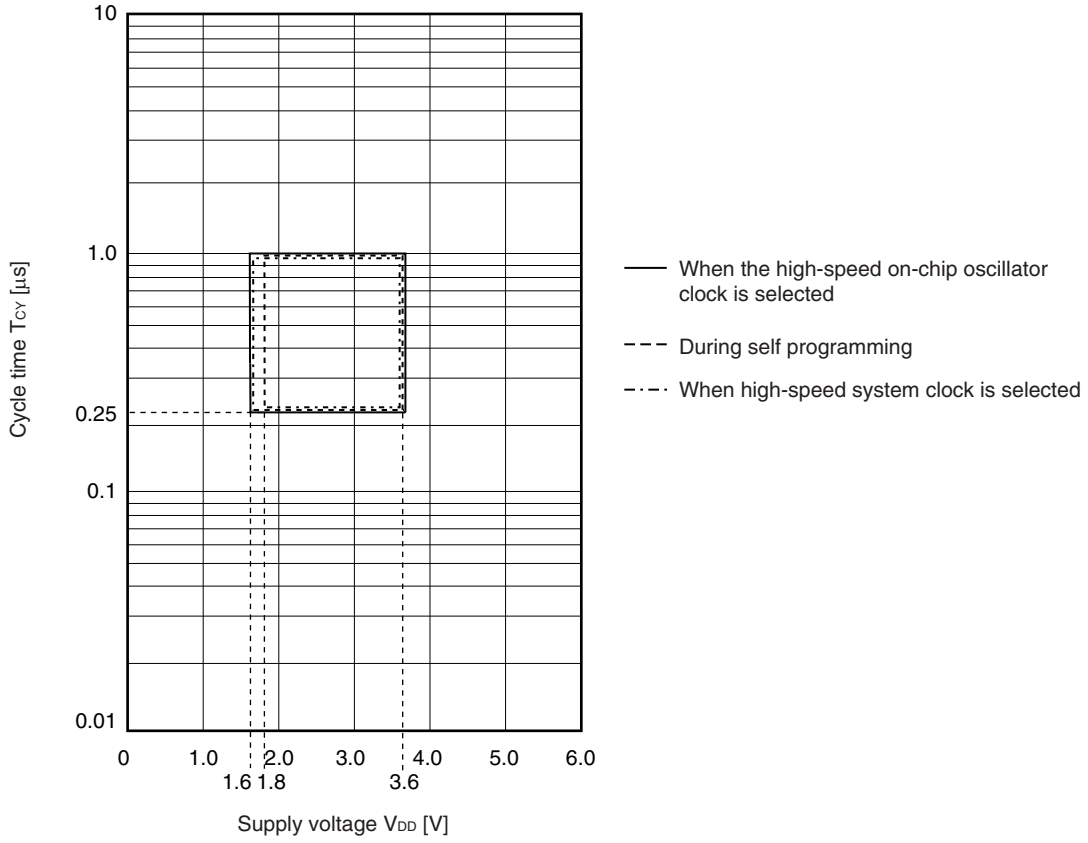
<R> $T_{CY}$ vs $V_{DD}$ (HS (high-speed main) mode)



<R> $T_{CY}$ vs $V_{DD}$ (LS (low-speed main) mode)

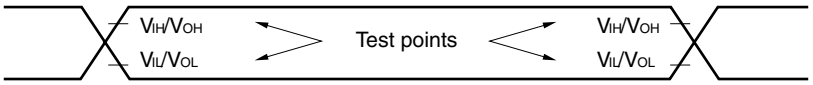



<R> $T_{CY}$ vs $V_{DD}$ (LV (low-voltage main) mode)

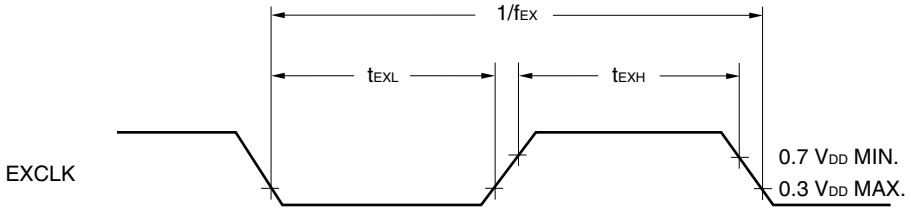


### AC Timing Test Points

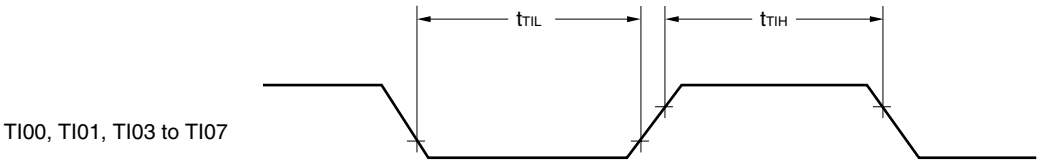
<R>

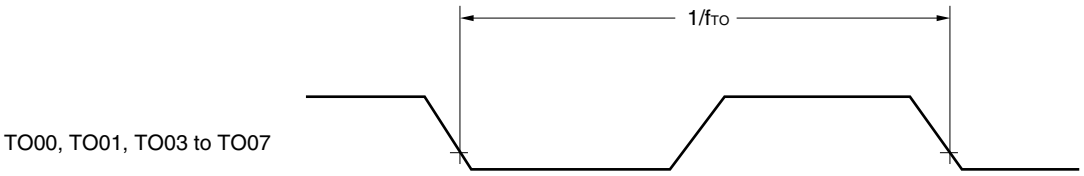



### External System Clock Timing

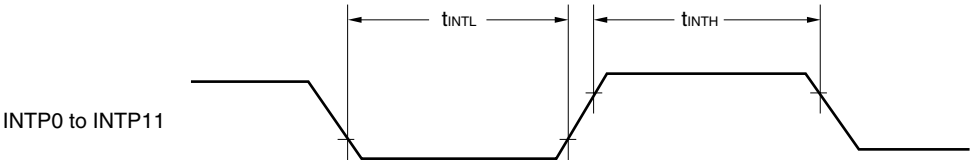



### <R> TI/TO Timing

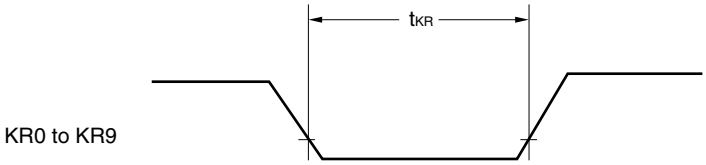


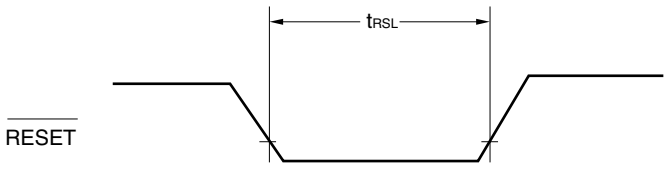

### Interrupt Request Input Timing




### Key Interrupt Input Timing



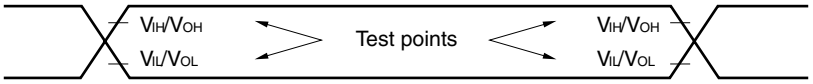
**$\overline{\text{RESET}}$ Input Timing**



## 29.5 Peripheral Functions Characteristics

### AC Timing Test Points

<R>




### 29.5.1 Serial array unit

#### (1) During communication at same potential (UART mode)

**($T_A$ = –40 to +85°C, 1.6 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | HS[Note 1] | | LS[Note 2] | | LV[Note 3] | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | MIN. | MAX. | MIN. | MAX. | MIN. | MAX. | |
| Transfer rate[Note 4] | | 2.4 V ≤ $EV_{DD}$ ≤ 3.6 V | | $f_{MCK}$/6 | | $f_{MCK}$/6 | | $f_{MCK}$/6 | bps |
| | | Theoretical value of the maximum transfer rate $f_{MCK}$ = $f_{CLK}$[Note 6] | | 5.3[Note 5] | | 1.3 | | 0.6 | Mbps |
| | | 1.8 V ≤ $EV_{DD}$ ≤ 3.6 V | | $f_{MCK}$/6 | | $f_{MCK}$/6 | | $f_{MCK}$/6 | bps |
| | | Theoretical value of the maximum transfer rate $f_{MCK}$ = $f_{CLK}$[Note 6] | | 5.3[Note 5] | | 1.3 | | 0.6 | Mbps |
| | | 1.7 V ≤ $EV_{DD}$ ≤ 3.6 V | | $f_{MCK}$/6 | | $f_{MCK}$/6 | | $f_{MCK}$/6 | bps |
| | | Theoretical value of the maximum transfer rate $f_{MCK}$ = $f_{CLK}$[Note 6] | | 5.3[Note 5] | | 1.3[Note 5] | | 0.6 | Mbps |
| | | 1.6 V ≤ $EV_{DD}$ ≤ 3.6 V | | – | | $f_{MCK}$/6 | | $f_{MCK}$/6 | bps |
| | | Theoretical value of the maximum transfer rate $f_{MCK}$ = $f_{CLK}$[Note 6] | | – | | 1.3[Note 5] | | 0.6 | Mbps |

**Notes 1.** HS is condition of HS (high-speed main) mode.

**2.** LS is condition of LS (low-speed main) mode.

**3.** LV is condition of LV (low-voltage main) mode.

**4.** Transfer rate in the SNOOZE mode is 4800 bps.

**5.** The following conditions are required for low-voltage interface when $EV_{DD0}$ < $V_{DD}$.

2.4 V ≤ $EV_{DD0}$ < 2.7 V : MAX. 2.6 Mbps

1.8 V ≤ $EV_{DD0}$ < 2.4 V : MAX. 1.3 Mbps

1.6 V ≤ $EV_{DD0}$ < 1.8 V : MAX. 0.6 Mbps

**6.** $f_{CLK}$ in each operating mode is as below.

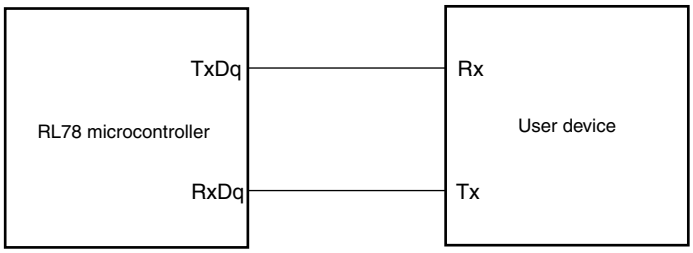
HS (high-speed main) mode: $f_{CLK}$ = 32 MHz

LS (low-speed main) mode: $f_{CLK}$ = 8 MHz

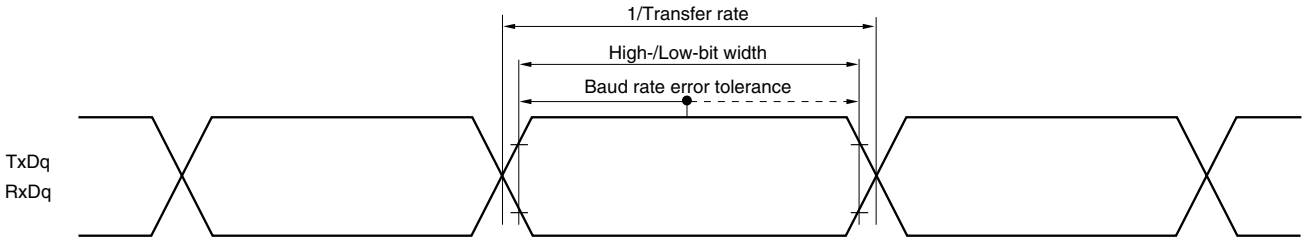
LV (low-voltage main) mode: $f_{CLK}$ = 4 MHz

**Caution Select the normal input buffer for the RxDq pin and the normal output mode for the TxDq pin by using port input mode register g (PIMg) and port output mode register g (POMg).**

**UART mode connection diagram (during communication at same potential)**




**UART mode bit width (during communication at same potential) (reference)**




**Remarks 1.** q: UART number (q = 0 to 2), g: PIM and POM number (g = 0, 1)

**2.** $f_{MCK}$: Serial array unit operation clock frequency
(Operation clock to be set by the CKSmn bit of serial mode register mn (SMRmn). m: Unit number, n: Channel number (mn = 00 to 03, 10, 11))

**(2) During communication at same potential (CSI mode) (master mode, SCKp... internal clock output, corresponding CSI00 only)**

**($T_A$ = −40 to +85°C, 2.7 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | | HS[Note 1] | | LS[Note 2] | | LV[Note 3] | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | MIN. | MAX. | MIN. | MAX. | MIN. | MAX. | |
| SCKp cycle time | $t_{KCY1}$ | 2.7 V ≤ $EV_{DD}$ ≤ 3.6 V | $t_{KCY1}$ ≥ 2/$f_{CLK}$ | 83.3 | | 250 | | 500 | | ns |
| SCKp high-/low-level width | $t_{KH1}$, $t_{KL1}$ | 2.7 V ≤ $EV_{DD}$ ≤ 3.6 V | | $t_{KCY1}$/2 −10 | | $t_{KCY1}$/2 −50 | | $t_{KCY1}$/2 −50 | | ns |
| SIp setup time (to SCKp↑)[Note 4] | $t_{SIK1}$ | 2.7 V ≤ $EV_{DD}$ ≤ 3.6 V | | 33 | | 110 | | 110 | | ns |
| SIp hold time (from SCKp↑)[Note 4] | $t_{KSI1}$ | 2.7 V ≤ $EV_{DD}$ ≤ 3.6 V | | 10 | | 10 | | 10 | | ns |
| Delay time from SCKp↓ to SOp output[Note 5] | $t_{KSO1}$ | C = 20 pF[Note 6] | | | 10 | | 10 | | 10 | ns |

**Notes 1.** HS is condition of HS (high-speed main) mode.

**2.** LS is condition of LS (low-speed main) mode.

**3.** LV is condition of LV (low-voltage main) mode.

**4.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The SIp setup time or SIp hold time becomes "from SCKp↓" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

**5.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The delay time to SOp output becomes "from SCKp↑" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

**6.** C is the load capacitance of the SCKp and SOp output lines.

**Caution Select the normal input buffer for the SIp pin and the normal output mode for the SOp pin and SCKp pin by using port input mode register g (PIMg) and port output mode register g (POMg).**

**Remarks 1.** p: CSI number (p = 00), m: Unit number (m = 0), n: Channel number (n = 0), g: PIM and POM numbers (g = 1)

**2.** $f_{MCK}$: Serial array unit operation clock frequency (Operation clock to be set by the CKSmn bit of serial mode register mn (SMRmn). m: Unit number, n: Channel number (mn = 00))

**(3) During communication at same potential (CSI mode) (master mode, SCKp... internal clock output)**

**($T_A$ = −40 to +85°C, 1.6 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | | HS[Note 1] | | LS[Note 2] | | LV[Note 3] | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | MIN. | MAX. | MIN. | MAX. | MIN. | MAX. | |
| SCKp cycle time | $t_{KCY2}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | $t_{KCY1}$ ≥ 4/$f_{CLK}$ | 125 | | 500 | | 1000 | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V | $t_{KCY1}$ ≥ 4/$f_{CLK}$ | 250 | | 500 | | 1000 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | $t_{KCY1}$ ≥ 4/$f_{CLK}$ | 500 | | 500 | | 1000 | | ns |
| | | 1.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | $t_{KCY1}$ ≥ 4/$f_{CLK}$ | 1000 | | 1000 | | 1000 | | ns |
| | | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V | $t_{KCY1}$ ≥ 4/$f_{CLK}$ | – | | 1000 | | 1000 | | ns |
| SCKp high-/low-level width | $t_{KH2}$, $t_{KL2}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | $t_{KCY2}$/2 −18 | | $t_{KCY2}$/2 −50 | | $t_{KCY2}$/2 −50 | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V | | $t_{KCY2}$/2 −38 | | $t_{KCY2}$/2 −50 | | $t_{KCY2}$/2 −50 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | | $t_{KCY2}$/2 −50 | | $t_{KCY2}$/2 −50 | | $t_{KCY2}$/2 −50 | | ns |
| | | 1.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | $t_{KCY2}$/2 −100 | | $t_{KCY2}$/2 −100 | | $t_{KCY2}$/2 −100 | | ns |
| | | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V | | – | | $t_{KCY2}$/2 −100 | | $t_{KCY2}$/2 −100 | | ns |
| SIp setup time (to SCKp↑)[Note 4] | $t_{SIK2}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 44 | | 110 | | 110 | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 75 | | 110 | | 110 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 110 | | 110 | | 110 | | ns |
| | | 1.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 220 | | 220 | | 220 | | ns |
| | | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V | | – | | 220 | | 220 | | ns |
| SIp hold time (from SCKp↑)[Note 4] | $t_{KSI2}$ | 1.7 V ≤ $EV_{DD}$ ≤ 3.6 V | | 19 | | 19 | | 19 | | ns |
| | | 1.6 V ≤ $EV_{DD}$ ≤ 3.6 V | | – | | 19 | | 19 | | ns |
| Delay time from SCKp↓ to SOp output[Note 5] | $t_{KSO2}$ | 1.7 V ≤ $EV_{DD}$ ≤ 3.6 V | C = 30 pF[Note 6] | | 25 | | 25 | | 25 | ns |
| | | 1.6 V ≤ $EV_{DD}$ ≤ 3.6 V | C = 30 pF[Note 6] | | – | | 25 | | 25 | ns |

**Notes 1.** HS is condition of HS (high-speed main) mode.

**2.** LS is condition of LS (low-speed main) mode.

**3.** LV is condition of LV (low-voltage main) mode.

**4.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The SIp setup time or SIp hold time becomes "from SCKp↓" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

**5.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The delay time to SOp output becomes "from SCKp↑" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

**6.** C is the load capacitance of the SCKp and SOp output lines.

**Caution Select the normal input buffer for the SIp pin and the normal output mode for the SOp pin and SCKp pin by using port input mode register g (PIMg) and port output mode register g (POMg).**

**Remark** p: CSI number (p = 00, 01, 10, 11, 20, 21), m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), g: PIM and POM numbers (g = 0, 1)

**(4) During communication at same potential (CSI mode) (slave mode, SCKp... external clock input)**

**($T_A$ = –40 to +85°C, 1.6 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | | HS[Note 1] | | LS[Note 2] | | LV[Note 3] | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | MIN. | MAX. | MIN. | MAX. | MIN. | MAX. | |
| SCKp cycle time[Note 4] | $t_{KCY2}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 16 MHz < $f_{MCK}$ | 8/$f_{MCK}$ | | – | | – | | ns |
| | | | $f_{MCK}$ ≤ 16 MHz | 6/$f_{MCK}$ | | 6/$f_{MCK}$ | | 6/$f_{MCK}$ | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 6/$f_{MCK}$ and 500ns | | 6/$f_{MCK}$ and 500ns | | 6/$f_{MCK}$ and 500ns | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 6/$f_{MCK}$ and 750ns | | 6/$f_{MCK}$ and 750ns | | 6/$f_{MCK}$ and 750ns | | ns |
| | | 1.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 6/$f_{MCK}$ and 1500ns | | 6/$f_{MCK}$ and 1500ns | | 6/$f_{MCK}$ and 1500ns | | ns |
| | | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V | | – | | 6/$f_{MCK}$ and 1500ns | | 6/$f_{MCK}$ and 1500ns | | ns |
| SCKp high-/low-level width | $t_{KH2}$, $t_{KL2}$ | 2.7 V ≤ $EV_{DD}$ ≤ 3.6 V | | $t_{KCY2}$/2 –8 | | $t_{KCY2}$/2 –8 | | $t_{KCY2}$/2 –8 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | | $t_{KCY2}$/2 –18 | | $t_{KCY2}$/2 –18 | | $t_{KCY2}$/2 –18 | | ns |
| | | 1.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | $t_{KCY2}$/2 –66 | | $t_{KCY2}$/2 –66 | | $t_{KCY2}$/2 –66 | | ns |
| | | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V | | – | | $t_{KCY2}$/2 –66 | | $t_{KCY2}$/2 –66 | | ns |
| SIp setup time (to SCKp↑)[Note 5] | $t_{SIK2}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 1/$f_{MCK}$ +20 | | 1/$f_{MCK}$ +30 | | 1/$f_{MCK}$ +30 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 1/$f_{MCK}$ +30 | | 1/$f_{MCK}$ +30 | | 1/$f_{MCK}$ +30 | | ns |
| | | 1.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 1/$f_{MCK}$ +40 | | 1/$f_{MCK}$ +40 | | 1/$f_{MCK}$ +40 | | ns |
| | | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V | | – | | 1/$f_{MCK}$ +40 | | 1/$f_{MCK}$ +40 | | ns |
| SIp hold time (from SCKp↑)[Note 5] | $t_{KSI2}$ | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 1/$f_{MCK}$ +31 | | 1/$f_{MCK}$ +31 | | 1/$f_{MCK}$ +31 | | ns |
| | | 1.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 1/$f_{MCK}$+ 250 | | 1/$f_{MCK}$+ 250 | | 1/$f_{MCK}$+ 250 | | ns |
| | | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V | | – | | 1/$f_{MCK}$+ 250 | | 1/$f_{MCK}$+ 250 | | ns |
| Delay time from SCKp↓ to SOp output[Note 6] | $t_{KSO2}$ | C = 30 pF[Note 7] | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 2/$f_{MCK}$ +44 | | 2/$f_{MCK}$ +110 | | 2/$f_{MCK}$ +110 | ns |
| | | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 2/$f_{MCK}$ +75 | | 2/$f_{MCK}$ +110 | | 2/$f_{MCK}$ +110 | ns |
| | | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 2/$f_{MCK}$ +110 | | 2/$f_{MCK}$ +110 | | 2/$f_{MCK}$ +110 | ns |
| | | | 1.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 2/$f_{MCK}$ +220 | | 2/$f_{MCK}$ +220 | | 2/$f_{MCK}$ +220 | ns |
| | | | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V | | – | | 2/$f_{MCK}$ +220 | | 2/$f_{MCK}$ +220 | ns |

(**Note, Caution** and **Remark** are listed on the next page.)

**Notes 1.** HS is condition of HS (high-speed main) mode.

**2.** LS is condition of LS (low-speed main) mode.

**3.** LV is condition of LV (low-voltage main) mode.

**4.** Transfer rate in the SNOOZE mode: MAX. 1 Mbps

**5.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The SIp setup time or SIp hold time becomes "from SCKp↓" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

**6.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The delay time to SOp output becomes "from SCKp↑" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

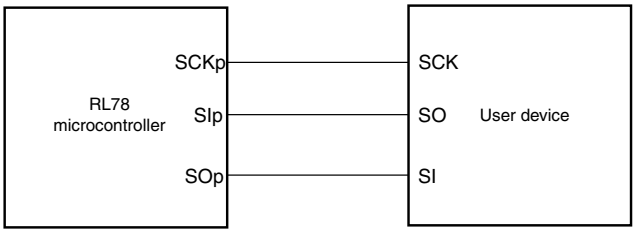
**7.** C is the load capacitance of the SOp output lines.

**Caution Select the normal input buffer for the SIp pin and SCKp pin and the normal output mode for the SOp pin by using port input mode register g (PIMg) and port output mode register g (POMg).**

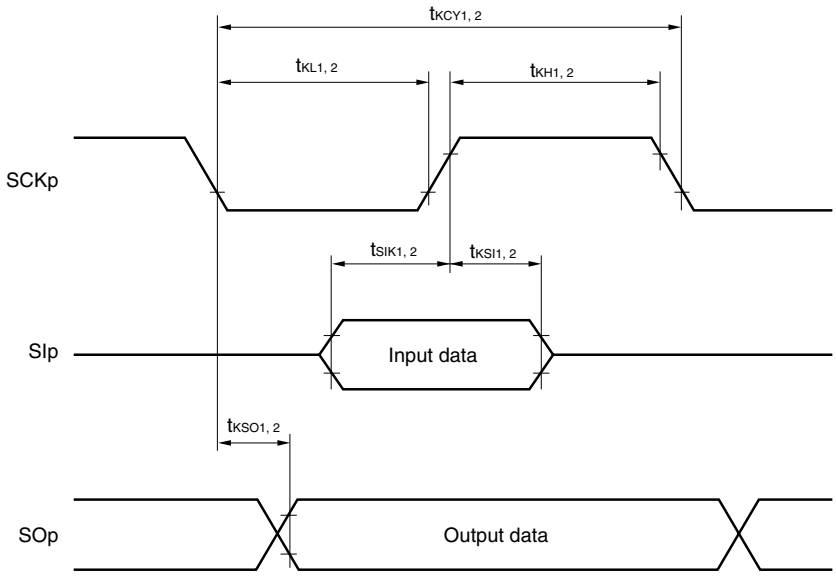
**Remarks 1.** p: CSI number (p = 00, 01, 10, 11, 20, 21), m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), g: PIM number (g = 0, 1)

**2.** $f_{MCK}$: Serial array unit operation clock frequency
(Operation clock to be set by the CKSmn bit of serial mode register mn (SMRmn). m: Unit number, n: Channel number (mn = 00 to 03, 10, 11))

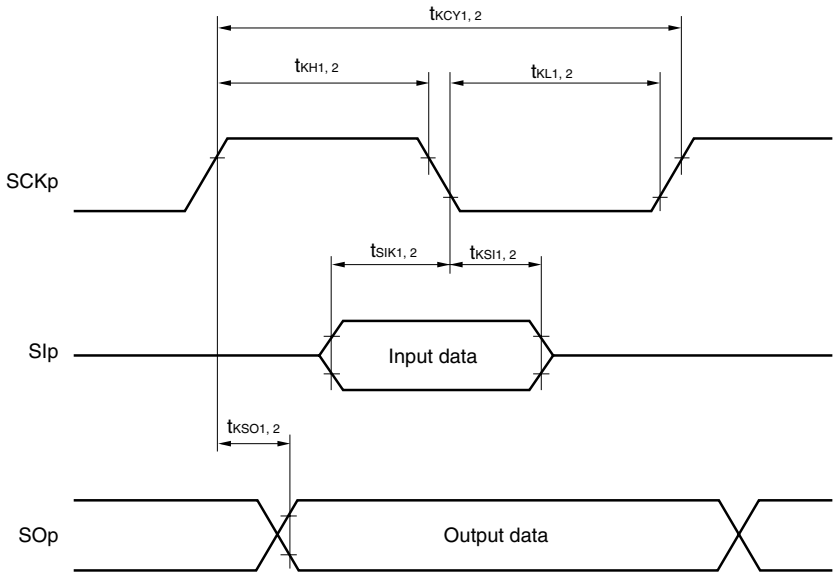
**CSI mode connection diagram (during communication at same potential)**




**CSI mode serial transfer timing (during communication at same potential)**
**(When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1.)**




**CSI mode serial transfer timing (during communication at same potential)**
**(When DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.)**




**Remarks 1.** p: CSI number (p = 00, 01, 10, 11, 20, 21)

**2.** m: Unit number, n: Channel number (mn = 00 to 03, 10, 11)

### (5) During communication at same potential (simplified $I^2C$ mode) (1/2)

**($T_A$ = –40 to +85°C, 1.6 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | HS[Note 1] | | LS[Note 2] | | LV[Note 3] | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | MIN. | MAX. | MIN. | MAX. | MIN. | MAX. | |
| SCLr clock frequency | $f_{SCL}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | | 1000[Note 4] | | 400[Note 4] | | 400[Note 4] | kHz |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V, $C_b$ = 100 pF, $R_b$ = 3 kΩ | | 400[Note 4] | | 400[Note 4] | | 400[Note 4] | kHz |
| | | 1.8 V ≤ $EV_{DD0}$ < 2.7 V, $C_b$ = 100 pF, $R_b$ = 5 kΩ | | 300[Note 4] | | 300[Note 4] | | 300[Note 4] | kHz |
| | | 1.7 V ≤ $EV_{DD0}$ < 1.8 V, $C_b$ = 100 pF, $R_b$ = 5 kΩ | | 250[Note 4] | | 250[Note 4] | | 250[Note 4] | kHz |
| | | 1.6 V ≤ $EV_{DD0}$ < 1.8 V, $C_b$ = 100 pF, $R_b$ = 5 kΩ | | – | | 250[Note 4] | | 250[Note 4] | kHz |
| Hold time when SCLr = “L” | $t_{LOW}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 475 | | 1150 | | 1150 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V, $C_b$ = 100 pF, $R_b$ = 3 kΩ | 1150 | | 1150 | | 1150 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ < 2.7 V, $C_b$ = 100 pF, $R_b$ = 5 kΩ | 1550 | | 1550 | | 1550 | | ns |
| | | 1.7 V ≤ $EV_{DD0}$ < 1.8 V, $C_b$ = 100 pF, $R_b$ = 5 kΩ | 1850 | | 1850 | | 1850 | | ns |
| | | 1.6 V ≤ $EV_{DD0}$ < 1.8 V, $C_b$ = 100 pF, $R_b$ = 5 kΩ | – | | 1850 | | 1850 | | ns |
| Hold time when SCLr = “H” | $t_{HIGH}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 475 | | 1150 | | 1150 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V, $C_b$ = 100 pF, $R_b$ = 3 kΩ | 1150 | | 1150 | | 1150 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ < 2.7 V, $C_b$ = 100 pF, $R_b$ = 5 kΩ | 1550 | | 1550 | | 1550 | | ns |
| | | 1.7 V ≤ $EV_{DD0}$ < 1.8 V, $C_b$ = 100 pF, $R_b$ = 5 kΩ | 1850 | | 1850 | | 1850 | | ns |
| | | 1.6 V ≤ $EV_{DD0}$ < 1.8 V, $C_b$ = 100 pF, $R_b$ = 5 kΩ | – | | 1850 | | 1850 | | ns |
| Data setup time (reception) | $t_{SU:DAT}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 1/$f_{MCK}$ + 85[Note 5] | | 1/$f_{MCK}$ + 145[Note 5] | | 1/$f_{MCK}$ + 145[Note 5] | | ns |
| | | 1.8 V ≤ $EV_{DD}$ ≤ 3.6 V, $C_b$ = 100 pF, $R_b$ = 3 kΩ | 1/$f_{MCK}$ + 145[Note 5] | | 1/$f_{MCK}$ + 145[Note 5] | | 1/$f_{MCK}$ + 145[Note 5] | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ < 2.7 V, $C_b$ = 100 pF, $R_b$ = 5 kΩ | 1/$f_{MCK}$+ 230[Note 5] | | 1/$f_{MCK}$+ 230[Note 5] | | 1/$f_{MCK}$+ 230[Note 5] | | ns |
| | | 1.7 V ≤ $EV_{DD}$ < 1.8 V, $C_b$ = 100 pF, $R_b$ = 5 kΩ | 1/$f_{MCK}$ + 290[Note 5] | | 1/$f_{MCK}$ + 290[Note 5] | | 1/$f_{MCK}$ + 290[Note 5] | | ns |
| | | 1.6 V ≤ $EV_{DD0}$ < 1.8 V, $C_b$ = 100 pF, $R_b$ = 5 kΩ | – | | 1/$f_{MCK}$ + 290[Note 5] | | 1/$f_{MCK}$ + 290[Note 5] | | ns |

(**Notes**, **Caution** and **Remarks** are listed on the next page.)

### (5) During communication at same potential (simplified I²C mode) (2/2)

**($T_A$ = −40 to +85°C, 1.6 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | HS[Note 1] | | LS[Note 2] | | LV[Note 3] | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | MIN. | MAX. | MIN. | MAX. | MIN. | MAX. | |
| Data hold time (transmission) | $t_{HD:DAT}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 0 | 305 | 0 | 305 | 0 | 305 | ns |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V, $C_b$ = 100 pF, $R_b$ = 3 kΩ | 0 | 355 | 0 | 355 | 0 | 355 | ns |
| | | 1.8 V ≤ $EV_{DD0}$ < 2.7 V, $C_b$ = 100 pF, $R_b$ = 5 kΩ | 0 | 405 | 0 | 405 | 0 | 405 | ns |
| | | 1.7 V ≤ $EV_{DD0}$ ≤ 1.8 V, $C_b$ = 100 pF, $R_b$ = 5 kΩ | 0 | 405 | 0 | 405 | 0 | 405 | ns |
| | | 1.6 V ≤ $EV_{DD0}$ < 1.8 V, $C_b$ = 100 pF, $R_b$ = 5 kΩ | – | – | 0 | 405 | 0 | 405 | ns |

**Notes 1.** HS is condition of HS (high-speed main) mode.

**2.** LS is condition of LS (low-speed main) mode.

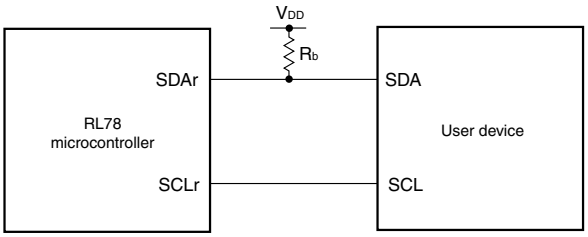
**3.** LV is condition of LV (low-voltage main) mode.

**4.** The value must also be $f_{CLK}$/4 or lower.

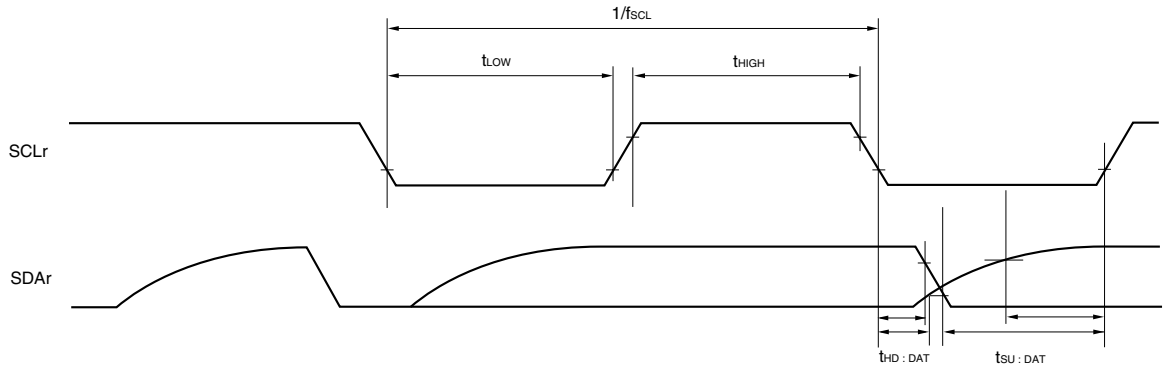
**5.** Set the $f_{MCK}$ value to keep the hold time of SCLr = "L" and SCLr = "H".

**Caution Select the normal input buffer and the N-ch open drain output ($V_{DD}$ tolerance (When 25- to 48-pin products)/$EV_{DD}$ tolerance (When 64-pin products)) mode for the SDAr pin and the normal output mode for the SCLr pin by using port input mode register g (PIMg) and port output mode register h (POMh).**

**Simplified I²C mode mode connection diagram (during communication at same potential)**




**Simplified I²C mode serial transfer timing (during communication at same potential)**




**Remarks 1.** $R_b$[Ω]: Communication line (SDAr) pull-up resistance, $C_b$[F]: Communication line (SDAr, SCLr) load capacitance

**2.** r: IIC number (r = 00, 01, 10, 11, 20, 21), g: PIM number (g = 0, 1), h: POM number (h = 0, 1)

**3.** $f_{MCK}$: Serial array unit operation clock frequency
(Operation clock to be set by the CKSmn bit of serial mode register mn (SMRmn). m: Unit number, n: Channel number, mn = 00 to 03, 10, 11)

**(6) Communication at different potential (1.8 V, 2.5 V) (UART mode) (dedicated baud rate generator output) (1/2)**

**($T_A$ = −40 to +85°C, 1.8 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | | | HS[Note 1] | | LS[Note 2] | | LV[Note 3] | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | MIN. | MAX. | MIN. | MAX. | MIN. | MAX. | |
| Transfer rate[Note 4] | | Reception | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V | | | $f_{MCK}$/6 | | $f_{MCK}$/6 | | $f_{MCK}$/6 | bps |
| | | | | Theoretical value of the maximum transfer rate $f_{MCK}$ = $f_{CLK}$[Note 7] | | 5.3 | | 1.3 | | 0.6 | Mbps |
| | | | 1.8 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 5] | | | $f_{MCK}$/6 | | $f_{MCK}$/6 | | $f_{MCK}$/6 | bps |
| | | | | Theoretical value of the maximum transfer rate $f_{MCK}$ = $f_{CLK}$[Note 7] | | 5.3 [Note 6] | | 1.3 | | 0.6 | Mbps |

**Notes 1.** HS is condition of HS (high-speed main) mode.

**2.** LS is condition of LS (low-speed main) mode.

**3.** LV is condition of LV (low-voltage main) mode.

**4.** Transfer rate in the SNOOZE mode is 4800 bps.

**5.** Use it with $EV_{DD0} \geq V_b$.

**6.** The following conditions are required for low-voltage interface when $EV_{DD0} < V_{DD}$.

2.4 V ≤ $EV_{DD0}$ < 2.7 V : MAX. 2.6 Mbps

1.8 V ≤ $EV_{DD0}$ < 2.4 V : MAX. 1.3 Mbps

**7.** $f_{CLK}$ in each operating mode is as below.

HS (high-speed main) mode: $f_{CLK}$ = 32 MHz

LS (low-speed main) mode: $f_{CLK}$ = 8 MHz

LV (low-voltage main) mode: $f_{CLK}$ = 4 MHz

**Caution Select the TTL input buffer for the RxDq pin and the N-ch open drain output ($V_{DD}$ tolerance (When 25- to 48-pin products)/$EV_{DD}$ tolerance (When 64-pin products)) mode for the TxDq pin by using port input** <R> **mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

**Remarks 1.** $V_b$[V]: Communication line voltage

**2.** q: UART number (q = 0 to 2), g: PIM and POM number (g = 0, 1)

**3.** $f_{MCK}$: Serial array unit operation clock frequency

(Operation clock to be set by the CKSmn bit of serial mode register mn (SMRmn). m: Unit number, n: Channel number (mn = 00 to 03, 10, 11)

**(6) Communication at different potential (1.8 V, 2.5 V) (UART mode) (dedicated baud rate generator output) (2/2)**

**($T_A$ = –40 to +85°C, 1.8 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | | | HS[Note 1] | | LS[Note 2] | | LV[Note 3] | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | MIN. | MAX. | MIN. | MAX. | MIN. | MAX. | |
| Transfer rate | | Transmission | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V | | | **Note 4** | | **Note 4** | | **Note 4** | bps |
| | | | | Theoretical value of the maximum transfer rate $C_b$ = 50 pF, $R_b$ = 2.7 kΩ, $V_b$ = 2.3 V | | 1.2 Note 5 | | 1.2 Note 5 | | 1.2 Note 5 | Mbps |
| | | | 1.8 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 6] | | | **Note 7** | | **Note 7** | | **Note 7** | bps |
| | | | | Theoretical value of the maximum transfer rate $C_b$ = 50 pF, $R_b$ = 5.5 kΩ, $V_b$ = 1.6 V | | 0.43 Note 8 | | 0.43 Note 8 | | 0.43 Note 8 | Mbps |

**Notes 1.** HS is condition of HS (high-speed main) mode.

**2.** LS is condition of LS (low-speed main) mode.

**3.** LV is condition of LV (low-voltage main) mode.

**4.** The smaller maximum transfer rate derived by using $f_{MCK}$/6 or the following expression is the valid maximum transfer rate.

Expression for calculating the transfer rate when 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V and 2.3 V ≤ $V_b$ ≤ 2.7 V

$$\text{Maximum transfer rate} = \frac{1}{\{-C_b \times R_b \times \ln(1 - \frac{2.0}{V_b})\} \times 3} \text{ [bps]}$$

$$\text{Baud rate error (theoretical value)} = \frac{\frac{1}{\text{Transfer rate} \times 2} - \{-C_b \times R_b \times \ln(1 - \frac{2.0}{V_b})\}}{(\frac{1}{\text{Transfer rate}}) \times \text{Number of transferred bits}} \times 100 \text{ [\%]}$$

* This value is the theoretical value of the relative difference between the transmission and reception sides.

**5.** This value as an example is calculated when the conditions described in the “Conditions” column are met. See **Note 4** above to calculate the maximum transfer rate under conditions of the customer.

**6.** Use it with $EV_{DD0} \geq V_b$.

**7.** The smaller maximum transfer rate derived by using $f_{MCK}$/6 or the following expression is the valid maximum transfer rate.

Expression for calculating the transfer rate when 1.8 V ≤ $EV_{DD0}$ < 3.3 V and 1.6 V ≤ $V_b$ ≤ 2.0 V

$$\text{Maximum transfer rate} = \frac{1}{\{-C_b \times R_b \times \ln(1 - \frac{1.5}{V_b})\} \times 3} \text{ [bps]}$$

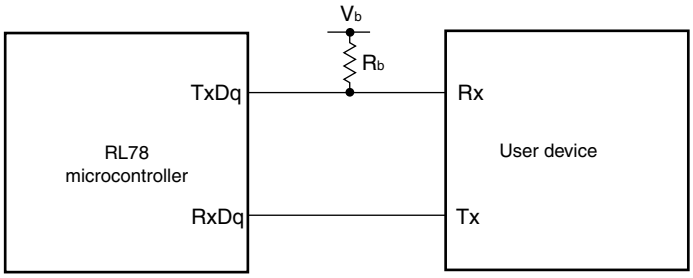
$$\text{Baud rate error (theoretical value)} = \frac{\frac{1}{\text{Transfer rate} \times 2} - \{-C_b \times R_b \times \ln(1 - \frac{1.5}{V_b})\}}{(\frac{1}{\text{Transfer rate}}) \times \text{Number of transferred bits}} \times 100 \text{ [\%]}$$

* This value is the theoretical value of the relative difference between the transmission and reception sides.

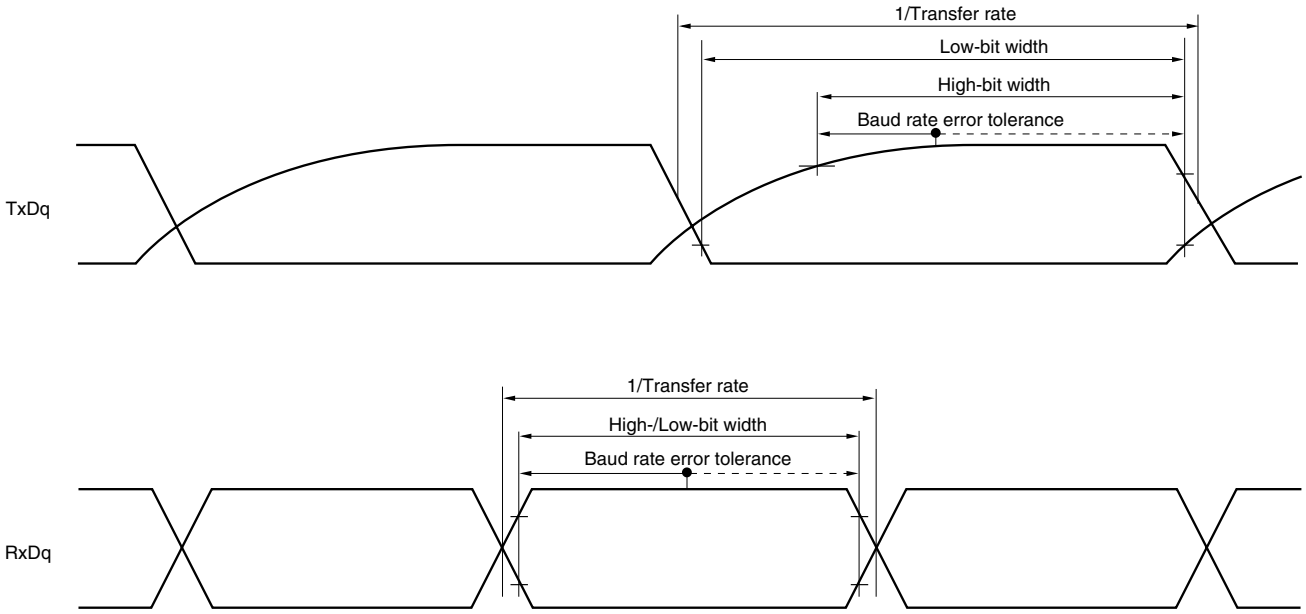
**8.** This value as an example is calculated when the conditions described in the “Conditions” column are met. See **Note 7** above to calculate the maximum transfer rate under conditions of the customer.

**Caution Select the TTL input buffer for the RxDq pin and the N-ch open drain output ($V_{DD}$ tolerance (When 25- to 48-pin products)/$EV_{DD}$ tolerance (When 64-pin products)) mode for the TxDq pin by using port input** <R> **mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

**UART mode connection diagram (during communication at different potential)**




**UART mode bit width (during communication at different potential) (reference)**




**Remarks 1.** $R_b$[Ω]: Communication line (TxDq) pull-up resistance,
$C_b$[F]: Communication line (TxDq) load capacitance, $V_b$[V]: Communication line voltage

**2.** q: UART number (q = 0 to 2), g: PIM and POM number (g = 0, 1)

**3.** $f_{MCK}$: Serial array unit operation clock frequency
(Operation clock to be set by the CKSmn bit of serial mode register mn (SMRmn).
m: Unit number, n: Channel number (mn = 00 to 03, 10, 11))

**(7) Communication at different potential (2.5 V) (CSI mode) (master mode, SCKp... internal clock output, corresponding CSI00 only)**

**($T_A$ = −40 to +85°C, 2.7 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | | HS[Note 1] | | LS[Note 2] | | LV[Note 3] | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | MIN. | MAX. | MIN. | MAX. | MIN. | MAX. | |
| SCKp cycle time | $t_{KCY1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 20 pF, $R_b$ = 2.7 kΩ | $t_{KCY1} \geq 2/f_{CLK}$ | 300 | | 1150 | | 1150 | | ns |
| SCKp high-level width | $t_{KH1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 20 pF, $R_b$ = 2.7 kΩ | | $t_{KCY1}$/2 − 120 | | $t_{KCY1}$/2 − 120 | | $t_{KCY1}$/2 − 120 | | ns |
| SCKp low-level width | $t_{KL1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 20 pF, $R_b$ = 2.7 kΩ | | $t_{KCY1}$/2 − 10 | | $t_{KCY1}$/2 − 50 | | $t_{KCY1}$/2 − 50 | | ns |
| SIp setup time (to SCKp↑)[Note 4] | $t_{SIK1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 20 pF, $R_b$ = 2.7 kΩ | | 121 | | 479 | | 479 | | ns |
| SIp hold time (from SCKp↑)[Note 4] | $t_{KSI1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 20 pF, $R_b$ = 2.7 kΩ | | 10 | | 10 | | 10 | | ns |
| Delay time from SCKp↓ to SOp output[Note 4] | $t_{KSO1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 20 pF, $R_b$ = 2.7 kΩ | | | 130 | | 130 | | 130 | ns |
| SIp setup time (to SCKp↓)[Note 5] | $t_{SIK1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 20 pF, $R_b$ = 2.7 kΩ | | 33 | | 110 | | 110 | | ns |
| SIp hold time (from SCKp↓)[Note 5] | $t_{KSI1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 20 pF, $R_b$ = 2.7 kΩ | | 10 | | 10 | | 10 | | ns |
| Delay time from SCKp↑ to SOp output[Note 5] | $t_{KSO1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 20 pF, $R_b$ = 2.7 kΩ | | | 10 | | 10 | | 10 | ns |

**Notes 1.** HS is condition of HS (high-speed main) mode.

**2.** LS is condition of LS (low-speed main) mode.

**3.** LV is condition of LV (low-voltage main) mode.

**4.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1.

**5.** When DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

**Caution Select the TTL input buffer for the SIp pin and the N-ch open drain output ($V_{DD}$ tolerance (When 25- to 48-pin products)/$EV_{DD}$ tolerance (When 64-pin products)) mode for the SOp pin and SCKp pin by using port input mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

<R>

**Remarks 1.** $R_b$[Ω]: Communication line (SCKp, SOp) pull-up resistance, $C_b$[F]: Communication line (SCKp, SOp) load capacitance, $V_b$[V]: Communication line voltage

**2.** p: CSI number (p = 00), m: Unit number (m = 0), n: Channel number (n = 0), g: PIM and POM number (g = 1)

**(8) Communication at different potential (1.8V, 2.5 V) (CSI mode) (master mode, SCKp... internal clock output) (1/2)**
**($T_A$ = −40 to +85°C, 1.8 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | | HS[Note 1] | | LS[Note 2] | | LV[Note 3] | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | MIN. | MAX. | MIN. | MAX. | MIN. | MAX. | |
| SCKp cycle time | $t_{KCY1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | $t_{KCY1} \geq 4/f_{CLK}$ | 500 | | 1150 | | 1150 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 4], $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | $t_{KCY1} \geq 4/f_{CLK}$ | 1150 | | 1150 | | 1150 | | ns |
| SCKp high-level width | $t_{KH1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | | $t_{KCY1}/2$ – 170 | | $t_{KCY1}/2$ – 170 | | $t_{KCY1}/2$ – 170 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 4], $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | | $t_{KCY1}/2$ – 458 | | $t_{KCY1}/2$ – 458 | | $t_{KCY1}/2$ – 458 | | ns |
| SCKp low-level width | $t_{KL1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | | $t_{KCY1}/2$ – 18 | | $t_{KCY1}/2$ – 50 | | $t_{KCY1}/2$ – 50 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 4], $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | | $t_{KCY1}/2$ – 50 | | $t_{KCY1}/2$ – 50 | | $t_{KCY1}/2$ – 50 | | ns |

**Notes 1.** HS is condition of HS (high-speed main) mode.
**2.** LS is condition of LS (low-speed main) mode.
**3.** LV is condition of LV (low-voltage main) mode.
**4.** Use it with $EV_{DD0} \geq V_b$.

**Caution Select the TTL input buffer for the SIp pin and the N-ch open drain output ($V_{DD}$ tolerance (When 25- to 48-pin products)/$EV_{DD}$ tolerance (When 64-pin products)) mode for the SOp pin and SCKp pin by using** <R> **port input mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

**Remarks 1.** $R_b$[Ω]: Communication line (SCKp, SOp) pull-up resistance, $C_b$[F]: Communication line (SCKp, SOp) load capacitance, $V_b$[V]: Communication line voltage
**2.** p: CSI number (p = 00, 10, 20), m: Unit number , n: Channel number (mn = 00, 02, 10), g: PIM and POM number (g = 0, 1)
**3.** CSI01, CSI11, and CSI21 cannot communicate at different potential. Use other CSI for communication at different potential.

**(8) Communication at different potential (1.8 V, 2.5 V) (CSI mode) (master mode, SCKp... internal clock output) (2/2)**

**($T_A$ = −40 to +85°C, 1.8 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | HS[Note 1] | | LS[Note 2] | | LV[Note 3] | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | MIN. | MAX. | MIN. | MAX. | MIN. | MAX. | |
| SIp setup time (to SCKp↑)[Note 4] | $t_{SIK1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | 177 | | 479 | | 479 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 6], $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | 479 | | 479 | | 479 | | ns |
| SIp hold time (from SCKp↑)[Note 4] | $t_{KSI1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | 19 | | 19 | | 19 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 6], $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | 19 | | 19 | | 19 | | ns |
| Delay time from SCKp↓ to SOp output[Note 4] | $t_{KSO1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | | 195 | | 195 | | 195 | ns |
| | | 1.8 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 6], $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | | 483 | | 483 | | 483 | ns |
| SIp setup time (to SCKp↓)[Note 5] | $t_{SIK1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | 44 | | 110 | | 110 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 6], $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | 110 | | 110 | | 110 | | ns |
| SIp hold time (from SCKp↓)[Note 5] | $t_{KSI1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | 19 | | 19 | | 19 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 6], $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | 19 | | 19 | | 19 | | ns |
| Delay time from SCKp↑ to SOp output[Note 5] | $t_{KSO1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | | 25 | | 25 | | 25 | ns |
| | | 1.8 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 6], $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | | 25 | | 25 | | 25 | ns |

**Notes 1.** HS is condition of HS (high-speed main) mode.

**2.** LS is condition of LS (low-speed main) mode.

**3.** LV is condition of LV (low-voltage main) mode.

**4.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1.

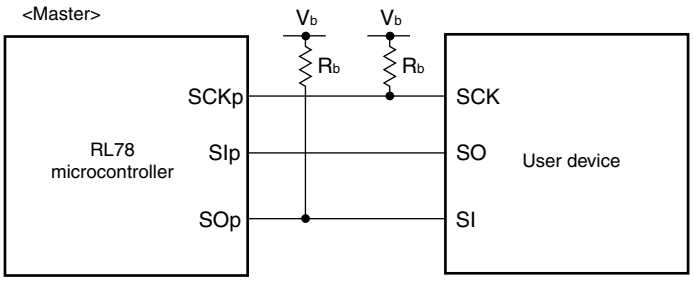
**5.** When DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

**6.** Use it with $EV_{DD0} \geq V_b$.

**Caution Select the TTL input buffer for the SIp pin and the N-ch open drain output ($V_{DD}$ tolerance (When 25- to 48-pin products)/$EV_{DD}$ tolerance (When 64-pin products)) mode for the SOp pin and SCKp pin by using** <R> **port input mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

**(Remarks** are listed on the next page.)

**CSI mode connection diagram (during communication at different potential)**

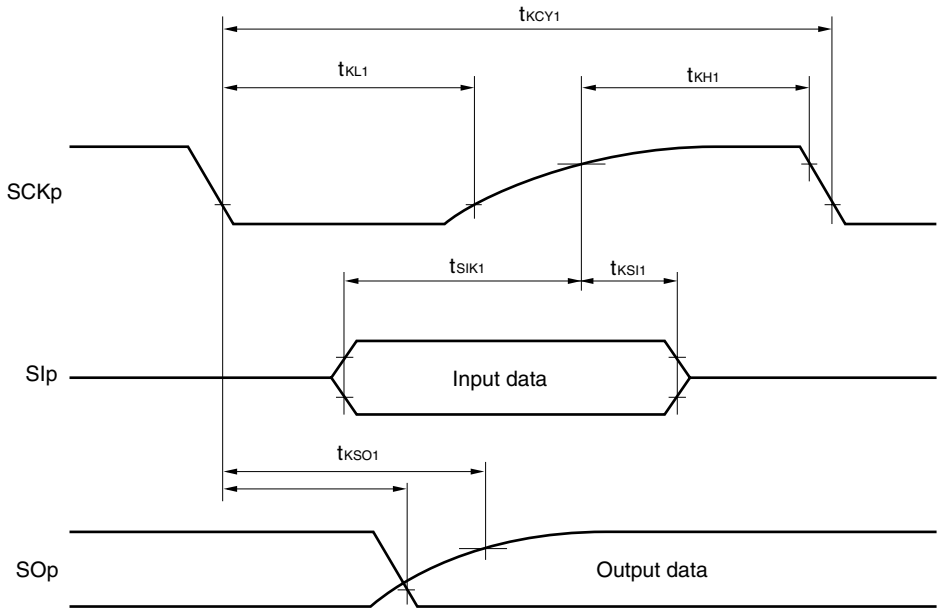



**Remarks 1.** $R_b$[Ω]: Communication line (SCKp, SOp) pull-up resistance, $C_b$[F]: Communication line (SCKp, SOp) load capacitance, $V_b$[V]: Communication line voltage

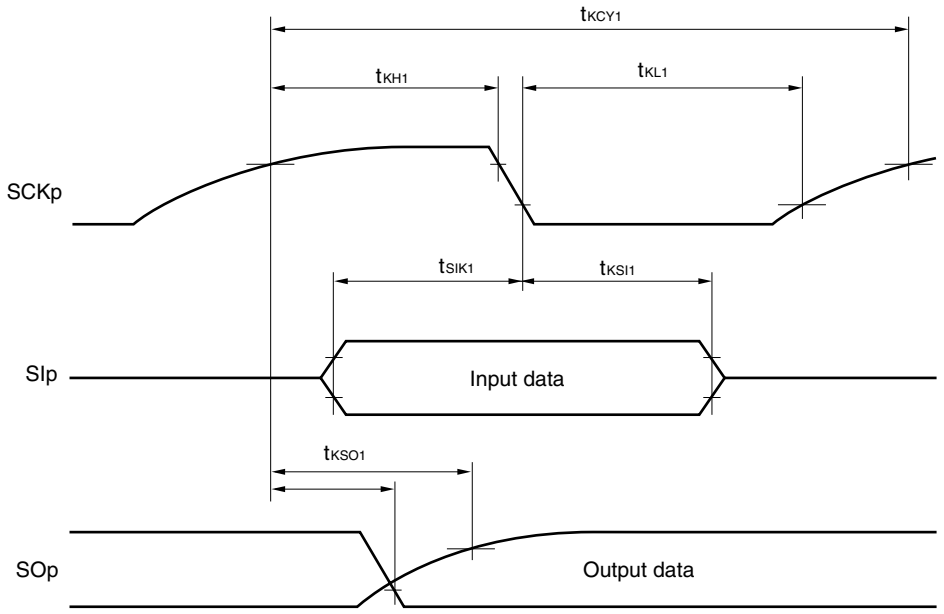
**2.** p: CSI number (p = 00, 10, 20), m: Unit number , n: Channel number (mn = 00, 02, 10), g: PIM and POM number (g = 0, 1)

**3.** CSI01, CSI11, and CSI21 cannot communicate at different potential. Use other CSI for communication at different potential.

**CSI mode serial transfer timing (master mode) (during communication at different potential)**
**(When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1.)**




**CSI mode serial transfer timing (master mode) (during communication at different potential)**
**(When DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.)**




**Remarks 1.** p: CSI number (p = 00, 10, 20), m: Unit number, n: Channel number (m = 00, 02, 10), g: PIM and POM number (g = 0, 1)

**2.** CSI01, CSI11, and CSI21 cannot communicate at different potential. Use other CSI for communication at different potential.

**(9) Communication at different potential (1.8 V, 2.5 V) (CSI mode) (slave mode, SCKp... external clock input)**

**($T_A$ = –40 to +85°C, 1.8 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | | HS[Note 1] | | LS[Note 2] | | LV[Note 3] | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | MIN. | MAX. | MIN. | MAX. | MIN. | MAX. | |
| SCKp cycle time[Note 4] | $t_{KCY2}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V | 24 MHz < $f_{MCK}$ | 20/$f_{MCK}$ | | – | | – | | ns |
| | | | 20 MHz < $f_{MCK}$ ≤ 24 MHz | 16/$f_{MCK}$ | | – | | – | | ns |
| | | | 16 MHz < $f_{MCK}$ ≤ 20 MHz | 14/$f_{MCK}$ | | – | | – | | ns |
| | | | 8 MHz < $f_{MCK}$ ≤ 16 MHz | 12/$f_{MCK}$ | | – | | – | | ns |
| | | | 4 MHz < $f_{MCK}$ ≤ 8 MHz | 8/$f_{MCK}$ | | 16/$f_{MCK}$ | | – | | ns |
| | | | $f_{MCK}$ ≤ 4 MHz | 6/$f_{MCK}$ | | 10/$f_{MCK}$ | | 10/$f_{MCK}$ | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 5] | 24 MHz < $f_{MCK}$ | 48/$f_{MCK}$ | | – | | – | | ns |
| | | | 20 MHz < $f_{MCK}$ ≤ 24 MHz | 36/$f_{MCK}$ | | – | | – | | ns |
| | | | 16 MHz < $f_{MCK}$ ≤ 20 MHz | 32/$f_{MCK}$ | | – | | – | | ns |
| | | | 8 MHz < $f_{MCK}$ ≤ 16 MHz | 26/$f_{MCK}$ | | – | | – | | ns |
| | | | 4 MHz < $f_{MCK}$ ≤ 8 MHz | 16/$f_{MCK}$ | | 16/$f_{MCK}$ | | – | | ns |
| | | | $f_{MCK}$ ≤ 4 MHz | 10/$f_{MCK}$ | | 10/$f_{MCK}$ | | 10/$f_{MCK}$ | | ns |
| SCKp high-/low-level width | $t_{KH2}$, $t_{KL2}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V | | $t_{KCY2}$/2 – 18 | | $t_{KCY2}$/2 – 50 | | $t_{KCY2}$/2 – 50 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 5] | | $t_{KCY2}$/2 – 50 | | $t_{KCY2}$/2 – 50 | | $t_{KCY2}$/2 – 50 | | ns |
| SIp setup time (to SCKp↑)[Note 6] | $t_{SIK2}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V | | 1/$f_{MCK}$ + 20 | | 1/$f_{MCK}$ + 30 | | 1/$f_{MCK}$ + 30 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 5] | | 1/$f_{MCK}$ + 30 | | 1/$f_{MCK}$ + 30 | | 1/$f_{MCK}$ + 30 | | ns |
| SIp hold time (from SCKp↑)[Note 6] | $t_{KSI2}$ | | | 1/$f_{MCK}$ + 31 | | 1/$f_{MCK}$ + 31 | | 1/$f_{MCK}$ + 31 | | ns |
| Delay time from SCKp↓ to SOp output[Note 7] | $t_{KSO2}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | | | 2/$f_{MCK}$ + 214 | | 2/$f_{MCK}$ + 573 | | 2/$f_{MCK}$ + 573 | ns |
| | | 1.8 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 5], $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | | | 2/$f_{MCK}$ + 573 | | 2/$f_{MCK}$ + 573 | | 2/$f_{MCK}$ + 573 | ns |

**Notes 1.** HS is condition of HS (high-speed main) mode.

**2.** LS is condition of LS (low-speed main) mode.

**3.** LV is condition of LV (low-voltage main) mode.

**4.** Transfer rate in the SNOOZE mode : MAX. 1 Mbps

**5.** Use it with $EV_{DD0}$ ≥ $V_b$.

**6.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The SIp setup time or SIp hold time becomes "from SCKp↓" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

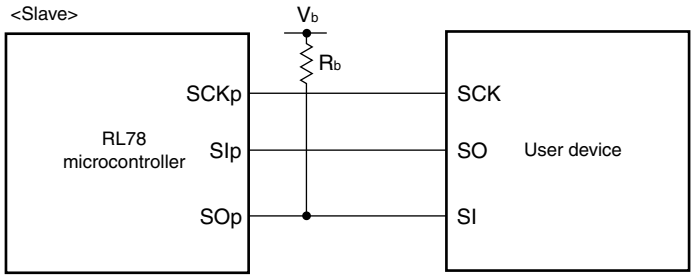
**7.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The delay time to SOp output becomes "from SCKp↑" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

**Caution Select the TTL input buffer for the SIp pin and SCKp pin and the N-ch open drain output ($V_{DD}$ tolerance (When 25- to 48-pin products)/$EV_{DD}$ tolerance (When 64-pin products)) mode for the SOp pin by using port input mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

<R>

(**Remarks** are listed on the next page.)

**CSI mode connection diagram (during communication at different potential)**



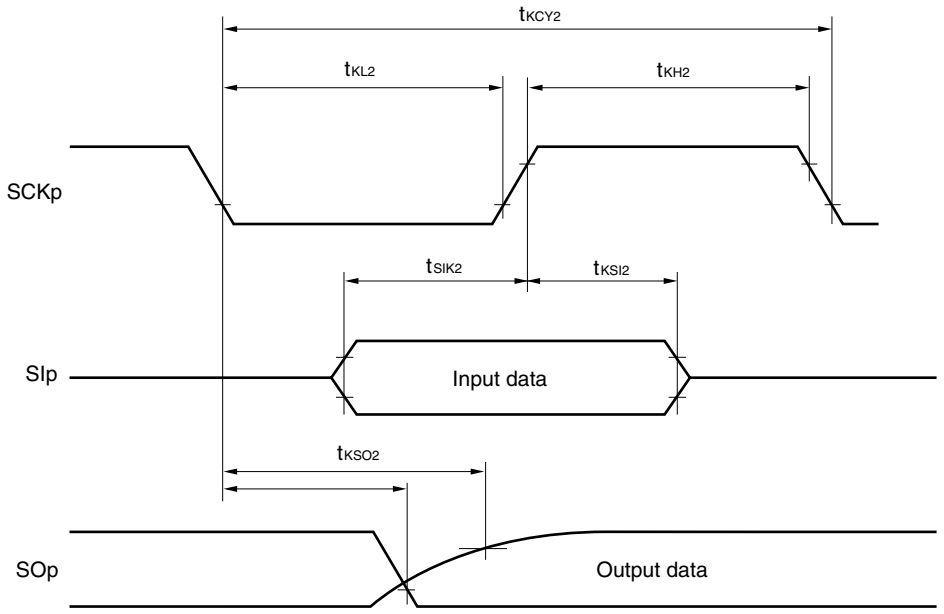

**Remarks 1.** $R_b$[Ω]: Communication line (SOp) pull-up resistance, $C_b$[F]: Communication line (SOp) load capacitance, $V_b$[V]: Communication line voltage

**2.** p: CSI number (p = 00, 10, 20), m: Unit number (m = 0, 1), n: Channel number (n = 00, 02, 10), g: PIM and POM number (g = 0, 1)

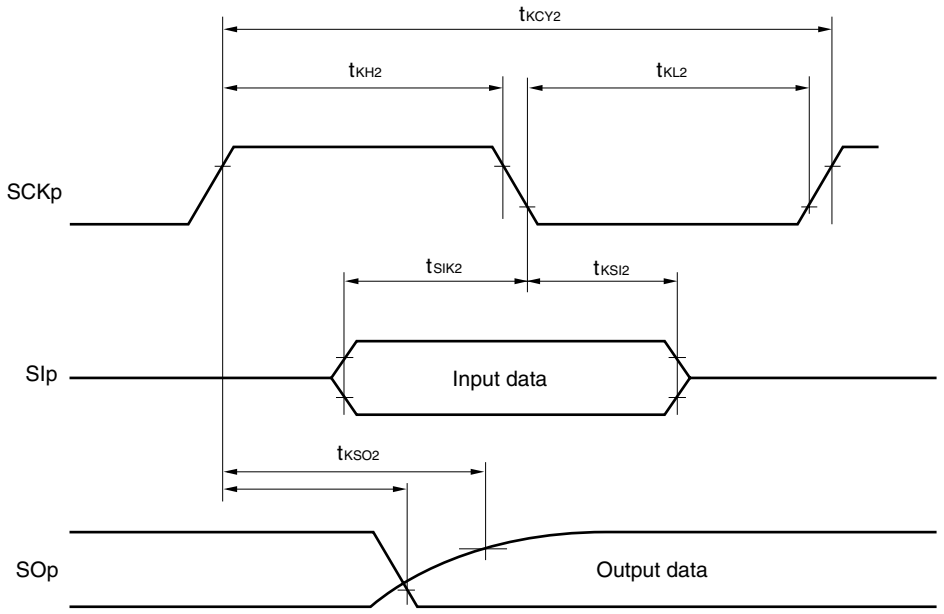
**3.** $f_{MCK}$: Serial array unit operation clock frequency
(Operation clock to be set by the CKSmn bit of serial mode register mn (SMRmn).
m: Unit number, n: Channel number (mn = 00, 02, 10))

**4.** CSI01, CSI11, and CSI21 cannot communicate at different potential. Use other CSI for communication at different potential.

**CSI mode serial transfer timing (slave mode) (during communication at different potential)**
**(When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1.)**




**CSI mode serial transfer timing (slave mode) (during communication at different potential)**
**(When DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.)**




**Remarks 1.** p: CSI number (p = 00, 10, 20), m: Unit number, n: Channel number (mn = 00, 02, 10), g: PIM and POM number (g = 0, 1)

**2.** CSI01, CSI11, and CSI21 cannot communicate at different potential. Use other CSI for communication at different potential.

**(10) Communication at different potential (1.8 V, 2.5 V) (simplified $I^2C$ mode) (1/2)**

**($T_A$ = –40 to +85°C, 1.8 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | HS[Note 1] | | LS[Note 2] | | LV[Note 3] | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | MIN. | MAX. | MIN. | MAX. | MIN. | MAX. | |
| SCLr clock frequency | $f_{SCL}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | | 1000[Note 4] | | 300[Note 4] | | 300[Note 4] | kHz |
| | | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 100 pF, $R_b$ = 2.7 kΩ | | 400[Note 4] | | 300[Note 4] | | 300[Note 4] | kHz |
| | | 1.8 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 5], $C_b$ = 100 pF, $R_b$ = 5.5 kΩ | | 300[Note 4] | | 300[Note 4] | | 300[Note 4] | kHz |
| Hold time when SCLr = “L” | $t_{LOW}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 475 | | 1550 | | 1550 | | ns |
| | | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 100 pF, $R_b$ = 2.7 kΩ | 1150 | | 1550 | | 1550 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 5], $C_b$ = 100 pF, $R_b$ = 5.5 kΩ | 1550 | | 1550 | | 1550 | | ns |
| Hold time when SCLr = “H” | $t_{HIGH}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 200 | | 610 | | 610 | | ns |
| | | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 100 pF, $R_b$ = 2.7 kΩ | 600 | | 610 | | 610 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 5], $C_b$ = 100 pF, $R_b$ = 5.5 kΩ | 610 | | 610 | | 610 | | ns |

(**Notes**, **Caution** and **Remarks** are listed on the next page.)

**(10) Communication at different potential (1.8 V, 2.5 V) (simplified $I^2C$ mode) (2/2)**

**($T_A$ = –40 to +85°C, 1.8 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | HS[Note 1] | | LS[Note 2] | | LV[Note 3] | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | MIN. | MAX. | MIN. | MAX. | MIN. | MAX. | |
| Data setup time (reception) | $t_{SU:DAT}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 1/$f_{MCK}$ + 135[Note 6] | | 1/$f_{MCK}$ + 190[Note 6] | | 1/$f_{MCK}$ + 190[Note 6] | | ns |
| | | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 100 pF, $R_b$ = 2.7 kΩ | 1/$f_{MCK}$ + 190[Note 6] | | 1/$f_{MCK}$ + 190[Note 6] | | 1/$f_{MCK}$ + 190[Note 6] | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 5], $C_b$ = 100 pF, $R_b$ = 5.5 kΩ | 1/$f_{MCK}$ + 190[Note 6] | | 1/$f_{MCK}$ + 190[Note 6] | | 1/$f_{MCK}$ + 190[Note 6] | | ns |
| Data hold time (transmission) | $t_{HD:DAT}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 0 | 305 | 0 | 305 | 0 | 305 | ns |
| | | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 100 pF, $R_b$ = 2.7 kΩ | 0 | 355 | 0 | 355 | 0 | 355 | ns |
| | | 1.8 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V[Note 5], $C_b$ = 100 pF, $R_b$ = 5.5 kΩ | 0 | 405 | 0 | 405 | 0 | 405 | ns |

**Notes 1.** HS is condition of HS (high-speed main) mode.

**2.** LS is condition of LS (low-speed main) mode.

**3.** LV is condition of LV (low-voltage main) mode.

**4.** The value must also be $f_{CLK}$/4 or lower.

**5.** Use it with $EV_{DD0} \geq V_b$.

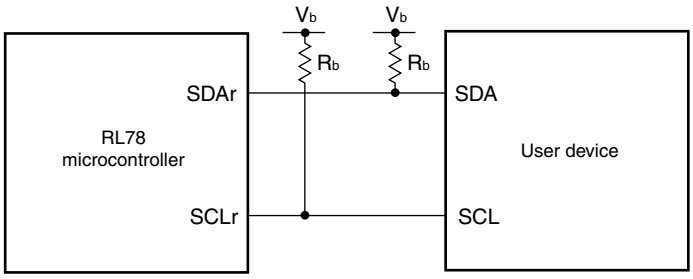
**6.** Set the $f_{MCK}$ value to keep the hold time of SCLr = “L” and SCLr = “H”.

**Caution Select the TTL input buffer and the N-ch open drain output ($V_{DD}$ tolerance (When 25- to 48-pin products)/$EV_{DD}$ tolerance (When 64-pin products)) mode for the SDAr pin and the N-ch open drain output ($V_{DD}$ tolerance (When 25- to 48-pin products)/$EV_{DD}$ tolerance (When 64-pin products)) mode for the SCLr pin by using port input mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

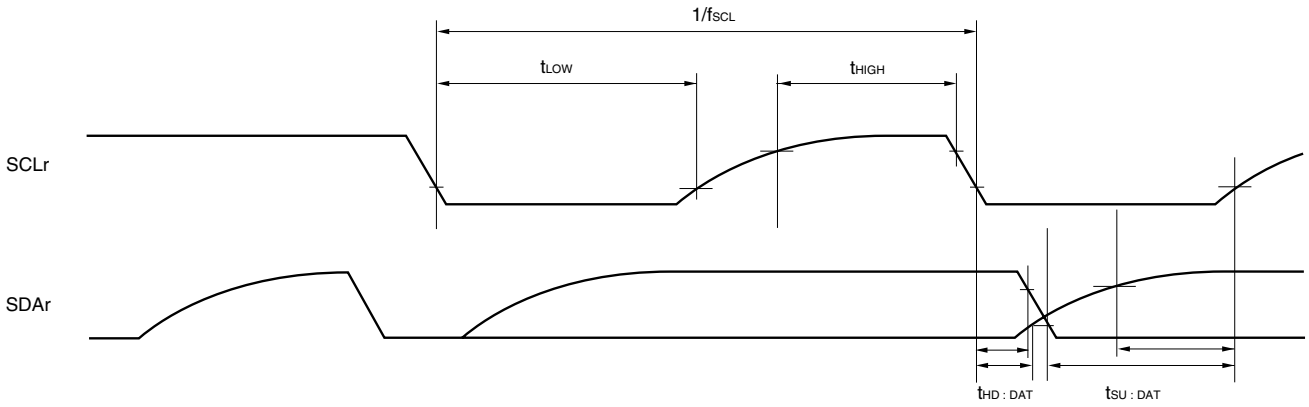
<R>

(**Remarks** are listed on the next page.)

**Simplified $I^2C$ mode connection diagram (during communication at different potential)**




**Simplified $I^2C$ mode serial transfer timing (during communication at different potential)**




**Remarks 1.** $R_b$[Ω]: Communication line (SDAr, SCLr) pull-up resistance, $C_b$[F]: Communication line (SDAr, SCLr) load capacitance, $V_b$[V]: Communication line voltage

**2.** r: IIC number (r = 00, 10, 20), g: PIM, POM number (g = 0, 1)

**3.** $f_{MCK}$: Serial array unit operation clock frequency
(Operation clock to be set by the CKSmn bit of serial mode register mn (SMRmn). m: Unit number, n: Channel number (mn = 00, 02, 10)

**4.** IIC01, IIC11, and IIC21 cannot communicate at different potential. Use IIC00, IIC10, or IIC20 for communication at different potential.

### 29.5.2 Serial interface IICA

#### (1) $I^2C$ standard mode

**($T_A$ = −40 to +85°C, 1.6 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | Standard Mode[Note 1] | | | | | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | | | HS[Note 2] | | LS[Note 3] | | LV[Note 4] | | |
| | | | MIN. | MAX. | MIN. | MIN. | MAX. | MIN. | |
| SCLA0 clock frequency | $f_{SCL}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 0 | 100 | 0 | 100 | 0 | 100 | kHz |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | 0 | 100 | 0 | 100 | 0 | 100 | |
| | | 1.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 0 | 100 | 0 | 100 | 0 | 100 | |
| | | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V | – | | 0 | 100 | 0 | 100 | |
| Setup time of restart condition | $t_{SU:STA}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 4.7 | | 4.7 | | 4.7 | | μs |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | 4.7 | | 4.7 | | 4.7 | | |
| | | 1.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 4.7 | | 4.7 | | 4.7 | | |
| | | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V | – | | 4.7 | | 4.7 | | |
| Hold time[Note 5] | $t_{HD:STA}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 4.0 | | 4.0 | | 4.0 | | μs |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | 4.0 | | 4.0 | | 4.0 | | |
| | | 1.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 4.0 | | 4.0 | | 4.0 | | |
| | | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V | – | | 4.0 | | 4.0 | | |
| Hold time when SCLA0 = “L” | $t_{LOW}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 4.7 | | 4.7 | | 4.7 | | μs |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | 4.7 | | 4.7 | | 4.7 | | |
| | | 1.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 4.7 | | 4.7 | | 4.7 | | |
| | | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V | – | | 4.7 | | 4.7 | | |
| Hold time when SCLA0 = “H” | $t_{HIGH}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 4.0 | | 4.0 | | 4.0 | | μs |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | 4.0 | | 4.0 | | 4.0 | | |
| | | 1.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 4.0 | | 4.0 | | 4.0 | | |
| | | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V | – | | 4.0 | | 4.0 | | |
| Data setup time (reception) | $t_{SU:DAT}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 250 | | 250 | | 250 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | 250 | | 250 | | 250 | | |
| | | 1.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 250 | | 250 | | 250 | | |
| | | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V | – | | 250 | | 250 | | |
| Data hold time (transmission)[Note 6] | $t_{HD:DAT}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 0 | 3.45 | 0 | 3.45 | 0 | 3.45 | μs |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | 0 | 3.45 | 0 | 3.45 | 0 | 3.45 | |
| | | 1.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 0 | 3.45 | 0 | 3.45 | 0 | 3.45 | |
| | | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V | – | – | 0 | 3.45 | 0 | 3.45 | |
| Setup time of stop condition | $t_{SU:STO}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 4.0 | | 4.0 | | 4.0 | | μs |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | 4.0 | | 4.0 | | 4.0 | | |
| | | 1.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 4.0 | | 4.0 | | 4.0 | | |
| | | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V | – | | 4.0 | | 4.0 | | |
| Bus-free time | $t_{BUF}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 4.7 | | 4.7 | | 4.7 | | μs |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | 4.7 | | 4.7 | | 4.7 | | |
| | | 1.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 4.7 | | 4.7 | | 4.7 | | |
| | | 1.6 V ≤ $EV_{DD0}$ ≤ 3.6 V | – | | 4.7 | | 4.7 | | |

(**Note** and **Remark** are listed on the next page.)

### (2) $I^2C$ fast mode, fast mode plus

**($T_A$ = −40 to +85°C, 1.6 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | Fast Mode[Note 7] | | | | | | Fast Mode Plus[Note 8] | | Unit |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | HS[Note 2] | | LS[Note 3] | | LV[Note 4] | | HS[Note 2] | | |
| | | | MIN. | MAX. | MIN. | MIN. | MAX. | MIN. | MAX. | MIN. | |
| SCLA0 clock frequency | $f_{SCL}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 0 | 400 | 0 | 400 | 0 | 400 | 0 | 1000 | kHz |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | 0 | 400 | 0 | 400 | 0 | 400 | – | | |
| Setup time of restart condition | $t_{SU:STA}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 0.6 | | 0.6 | | 0.6 | | 0.26 | | μs |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | 0.6 | | 0.6 | | 0.6 | | – | | |
| Hold time[Note 5] | $t_{HD:STA}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 0.6 | | 0.6 | | 0.6 | | 0.26 | | μs |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | 0.6 | | 0.6 | | 0.6 | | – | | |
| Hold time when SCLA0 = “L” | $t_{LOW}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 1.3 | | 1.3 | | 1.3 | | 0.5 | | μs |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | 1.3 | | 1.3 | | 1.3 | | – | | |
| Hold time when SCLA0 = “H” | $t_{HIGH}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 0.6 | | 0.6 | | 0.6 | | 0.26 | | μs |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | 0.6 | | 0.6 | | 0.6 | | – | | |
| Data setup time (reception) | $t_{SU:DAT}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 100 | | 100 | | 100 | | 50 | | ns |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | 100 | | 100 | | 100 | | – | | |
| Data hold time (transmission)[Note 6] | $t_{HD:DAT}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 0 | 0.9 | 0 | 0.9 | 0 | 0.9 | 0 | 450 | μs |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | 0 | 0.9 | 0 | 0.9 | 0 | 0.9 | – | | |
| Setup time of stop condition | $t_{SU:STO}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 0.6 | | 0.6 | | 0.6 | | 0.26 | | μs |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | 0.6 | | 0.6 | | 0.6 | | – | | |
| Bus-free time | $t_{BUF}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 1.3 | | 1.3 | | 1.3 | | 0.5 | | μs |
| | | 1.8 V ≤ $EV_{DD0}$ ≤ 3.6 V | 1.3 | | 1.3 | | 1.3 | | – | | |

**Notes 1.** In normal mode, use it with $f_{CLK}$ ≥ 1 MHz, 1.6 V ≤ $EV_{DD}$ ≤ 3.6 V.

**2.** HS is condition of HS (high-speed main) mode.

**3.** LS is condition of LS (low-speed main) mode.

**4.** LV is condition of LV (low-voltage main) mode.

**5.** The first clock pulse is generated after this period when the start/restart condition is detected.

**6.** The maximum value (MAX.) of $t_{HD:DAT}$ is during normal transfer and a wait state is inserted in the $\overline{ACK}$ (acknowledge) timing.

**7.** In fast mode, use it with $f_{CLK}$ ≥ 3.5 MHz, 1.8 V ≤ $EV_{DD}$ ≤ 3.6 V.

**8.** In fast mode plus, use it with $f_{CLK}$ ≥ 10 MHz, 2.7 V ≤ $EV_{DD}$ ≤ 3.6 V.

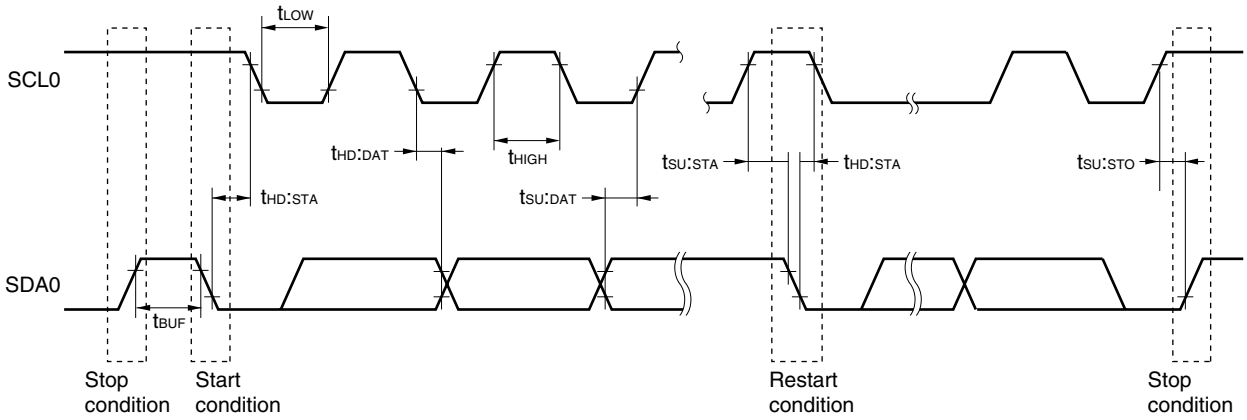
**Remark** The maximum value of $C_b$ (communication line capacitance) and the value of $R_b$ (communication line pull-up resistor) at that time in each mode are as follows.

Standard mode: $C_b$ = 400 pF, $R_b$ = 2.7 kΩ

Fast mode: $C_b$ = 320 pF, $R_b$ = 1.1 kΩ

Fast mode plus: $C_b$ = 120 pF, $R_b$ = 1.1 kΩ

**IICA serial transfer timing**



## 29.6 Analog Characteristics

### 29.6.1 A/D converter characteristics

**Division of A/D Converter Characteristics**

| Reference voltag / Input channel | Reference voltage (+) = $AV_{REFP}$<br>Reference voltage (−) = $AV_{REFM}$ | Reference voltage (+) = $AV_{DD}$<br>Reference voltage (−) = $AV_{SS}$ | Reference voltage (+) = Internal refrence voltage<br>Reference voltage (−) = $AV_{SS}$ |
|---|---|---|---|
| High-accuracy channel; ANI0 to ANI12<br>(input buffer power supply: $AV_{DD}$) | See **29.6.1 (1)**<br>See **29.6.1 (2)** | See **29.6.1 (3)** | See **29.6.1 (6)** |
| Standard channel; ANI16 to ANI30<br>(input buffer power supply: $V_{DD}$ or $EV_{DD0}$) | See **29.6.1 (4)** | See **29.6.1 (5)** | |
| Temperature sensor, internal reference voltage output | See **29.6.1 (4)** | See **29.6.1 (5)** | − |

<R> **(1) When reference voltage (+) = $AV_{REFP}$/ANI0 (ADREFP1 = 0, ADREFP0 = 1), reference voltage (−) = $AV_{REFM}$/ANI1 (ADREFM = 1), target for conversion: ANI2 to ANI12**

**($T_A$ = −40 to +85°C, 2.7 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = 0 V, $AV_{SS}$ = 0 V, Reference voltage (+) = $AV_{REFP}$, Reference voltage (−) = $AV_{REFM}$ = 0 V, HALT mode)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Resolution | $R_{ES}$ | | | | 12 | bit |
| Overall error[Notes 1, 2, 3] | AINL | 12-bit resolution | | ±1.7 | ±3.3 | LSB |
| Conversion time | $t_{CONV}$ | ADTYP = 0, 12-bit resolution | 3.375 | | | μs |
| Zero-scale error[Notes 1, 2, 3] | $E_{ZS}$ | 12-bit resolution | | ±1.3 | ±3.2 | LSB |
| Full-scale error[Notes 1, 2, 3] | $E_{FS}$ | 12-bit resolution | | ±0.7 | ±2.9 | LSB |
| Integral linearity error[Notes 1, 2, 3] | ILE | 12-bit resolution | | ±1.0 | ±1.4 | LSB |
| Differential linearity error[Notes 1, 2, 3] | DLE | 12-bit resolution | | ±0.9 | ±1.2 | LSB |
| Analog input voltage | $V_{AIN}$ | | 0 | | $AV_{REFP}$ | V |

**Notes 1.** TYP. Value is the average value at $AV_{DD}$ = $AV_{REFP}$ = 3 V and $T_A$ = 25°C. MAX. value is the average value ±3σ at normalized distribution.

**2.** These values are the results of characteristic evaluation and are not checked for shipment.

**3.** Excludes quantization error (±1/2 LSB).

**Cautions 1. Route the wiring so that noise will not be superimposed on each power line and ground line, and insert a capacitor to suppress noise.**
**In addition, separate the reference voltage line of $AV_{REFP}$ from the other power lines to keep it free from the influences of noise.**

**2. During A/D conversion, keep a pulse, such as a digital signal, that abruptly changes its level from being input to or output from the pins adjacent to the converter pins and P20 to P27 and P150 to P154.**

<R> **(2) When reference voltage (+) = $AV_{REFP}$/ANI0 (ADREFP1 = 0, ADREFP0 = 1), reference voltage (−) = $AV_{REFM}$/ANI1 (ADREFM = 1), target for conversion: ANI2 to ANI12**

**($T_A$ = −40 to +85°C, 1.6 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = 0 V, $AV_{SS}$ = 0 V, Reference voltage (+) = $AV_{REFP}$, Reference voltage (−) = $AV_{REFM}$ = 0 V)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Resolution | $R_{ES}$ | | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 8 | | 12 | bit |
| | | | 1.8 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 8 | | 10 Note 1 | |
| | | | 1.6 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 8 Note 2 | | | |
| Overall error Note 3 | AINL | 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±6.0 | LSB |
| | | 10-bit resolution | 1.8 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±5.0 | |
| | | 8-bit resolution | 1.6 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.5 | |
| Conversion time | $t_{CONV}$ | ADTYP = 0, 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 3.375 | | | μs |
| | | ADTYP = 0, 10-bit resolution Note 1 | 1.8 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 6.75 | | | |
| | | ADTYP = 0, 8-bit resolution Note 2 | 1.6 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 13.5 | | | |
| | | ADTYP = 1, 8-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 2.5625 | | | |
| | | | 1.8 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 5.125 | | | |
| | | | 1.6 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 10.25 | | | |
| Zero-scale error Note 3 | $E_{ZS}$ | 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±4.5 | LSB |
| | | 10-bit resolution | 1.8 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±4.5 | |
| | | 8-bit resolution | 1.6 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.0 | |
| Full-scale error Note 3 | $E_{FS}$ | 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±4.5 | LSB |
| | | 10-bit resolution | 1.8 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±4.5 | |
| | | 8-bit resolution | 1.6 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.0 | |
| Integral linearity error Note 3 | ILE | 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.0 | LSB |
| | | 10-bit resolution | 1.8 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±1.5 | |
| | | 8-bit resolution | 1.6 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±1.0 | |
| Differential linearity error Note 3 | DLE | 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±1.5 | LSB |
| | | 10-bit resolution | 1.8 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±1.5 | |
| | | 8-bit resolution | 1.6 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±1.0 | |
| Analog input voltage | $V_{AIN}$ | | | 0 | | $AV_{REFP}$ | V |

**Notes 1.** Cannot be used for lower 2 bit of ADCR register

**2.** Cannot be used for lower 4 bit of ADCR register

**3.** Excludes quantization error (±1/2 LSB).

<R> **(3) When reference voltage (+) = $AV_{DD}$ (ADREFP1 = 0, ADREFP0 = 0), reference voltage (−) = $AV_{SS}$ (ADREFM = 0), target for conversion: ANI0 to ANI12**

**($T_A$ = −40 to +85°C, 1.6 V ≤ $AV_{DD}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = 0 V, $AV_{SS}$ = 0 V, Reference voltage (+) = $AV_{DD}$, Reference voltage (−) = $AV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Resolution | $R_{ES}$ | | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | 8 | | 12 | bit |
| | | | 1.8 V ≤ $AV_{DD}$ ≤ 3.6 V | 8 | | 10 Note 1 | |
| | | | 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V | 8 Note 2 | | | |
| Overall error Note 3 | AINL | 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±7.5 | LSB |
| | | 10-bit resolution | 1.8 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±5.5 | |
| | | 8-bit resolution | 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±3.0 | |
| Conversion time | $t_{CONV}$ | ADTYP = 0, 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | 3.375 | | | μs |
| | | ADTYP = 0, 10-bit resolution Note 1 | 1.8 V ≤ $AV_{DD}$ ≤ 3.6 V | 6.75 | | | |
| | | ADTYP = 0, 8-bit resolution Note 2 | 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V | 13.5 | | | |
| | | ADTYP = 1, 8-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | 2.5625 | | | |
| | | | 1.8 V ≤ $AV_{DD}$ ≤ 3.6 V | 5.125 | | | |
| | | | 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V | 10.25 | | | |
| Zero-scale error Note 3 | $E_{ZS}$ | 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±6.0 | LSB |
| | | 10-bit resolution | 1.8 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±5.0 | |
| | | 8-bit resolution | 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.5 | |
| Full-scale error Note 3 | $E_{FS}$ | 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±6.0 | LSB |
| | | 10-bit resolution | 1.8 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±5.0 | |
| | | 8-bit resolution | 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.5 | |
| Integral linearity error Note 3 | ILE | 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±3.0 | LSB |
| | | 10-bit resolution | 1.8 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.0 | |
| | | 8-bit resolution | 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±1.5 | |
| Differential linearity error Note 3 | DLE | 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.0 | LSB |
| | | 10-bit resolution | 1.8 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.0 | |
| | | 8-bit resolution | 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±1.5 | |
| <R> Analog input voltage | $V_{AIN}$ | | | 0 | | $AV_{DD}$ | V |

**Notes 1.** Cannot be used for lower 2 bit of ADCR register

**2.** Cannot be used for lower 4 bit of ADCR register

**3.** Excludes quantization error (±1/2 LSB).

<R> **(4) When reference voltage (+) = $AV_{REFP}$/ANI0 (ADREFP1 = 0, ADREFP0 = 1), reference voltage (–) = $AV_{REFM}$/ANI1 (ADREFM = 1), target for conversion: ANI16 to ANI30, interanal reference voltage, temperature sensor output voltage**

**($T_A$ = −40 to +85°C, 1.6 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, 1.6 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V, $AV_{SS}$ = 0 V, Reference voltage (+) = $AV_{REFP}$, Reference voltage (–) = $AV_{REFM}$ = 0 V)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Resolution | $R_{ES}$ | | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 8 | | 12 | bit |
| | | | 1.8 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 8 | | 10 Note 1 | |
| | | | 1.6 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 8 Note 2 | | | |
| Overall error Note 3 | AINL | 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±7.0 | LSB |
| | | 10-bit resolution | 1.8 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±5.5 | |
| | | 8-bit resolution | 1.6 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±3.0 | |
| Conversion time | $t_{CONV}$ | ADTYP = 0, 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 4.125 | | | μs |
| | | ADTYP = 0, 10-bit resolution Note 1 | 1.8 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 9.5 | | | |
| | | ADTYP = 0, 8-bit resolution Note 2 | 1.6 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 57.5 | | | |
| | | ADTYP = 1, 8-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 3.3125 | | | |
| | | | 1.8 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 7.875 | | | |
| | | | 1.6 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 54.25 | | | |
| Zero-scale error Note 3 | $E_{ZS}$ | 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±5.0 | LSB |
| | | 10-bit resolution | 1.8 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±5.0 | |
| | | 8-bit resolution | 1.6 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.5 | |
| Full-scale error Note 3 | $E_{FS}$ | 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±5.0 | LSB |
| | | 10-bit resolution | 1.8 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±5.0 | |
| | | 8-bit resolution | 1.6 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.5 | |
| Integral linearity error Note 3 | ILE | 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±3.0 | LSB |
| | | 10-bit resolution | 1.8 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.0 | |
| | | 8-bit resolution | 1.6 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±1.5 | |
| Differential linearity error Note 3 | DLE | 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.0 | LSB |
| | | 10-bit resolution | 1.8 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.0 | |
| | | 8-bit resolution | 1.6 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±1.5 | |
| Analog input voltage | $V_{AIN}$ | | | 0 | | $AV_{REFP}$ and $EV_{DD0}$ | V |
| | | Interanal reference voltage (2.4 V ≤ $V_{DD}$ ≤ 3.6 V, HS (high-speed main) mode) | | $V_{BGR}$ Note 4 | | | V |
| | | Temperature sensor output voltage (2.4 V ≤ $V_{DD}$ ≤ 3.6 V, HS (high-speed main) mode) | | $V_{TMPS25}$ Note 4 | | | V |

**Notes 1.** Cannot be used for lower 2 bit of ADCR register

**2.** Cannot be used for lower 4 bit of ADCR register

**3.** Excludes quantization error (±1/2 LSB).

**4.** See **29.6.2 Temperature sensor, internal reference voltage output characteristics**.

<R> **(5) When reference voltage (+) = $AV_{DD}$ (ADREFP1 = 0, ADREFP0 = 0), reference voltage (–) = $AV_{SS}$ (ADREFM = 0), target for conversion: ANI16 to ANI30, interanal reference voltage, temperature sensor output voltage**

**($T_A$ = −40 to +85°C, 1.6 V ≤ $EV_{DD0}$ ≤ $V_{DD0}$ ≤ 3.6 V, 1.6 V ≤ $AV_{DD}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V, $AV_{SS}$ = 0 V, Reference voltage (+) = $AV_{DD}$, Reference voltage (–) = $AV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Resolution | RES | | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | 8 | | 12 | bit |
| | | | 1.8 V ≤ $AV_{DD}$ ≤ 3.6 V | 8 | | 10 Note 1 | |
| | | | 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V | 8 Note 2 | | | |
| Overall error Note 3 | AINL | 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±8.5 | LSB |
| | | 10-bit resolution | 1.8 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±6.0 | |
| | | 8-bit resolution | 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±3.5 | |
| Conversion time | tCONV | ADTYP = 0, 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | 4.125 | | | μs |
| | | ADTYP = 0, 10-bit resolution Note 1 | 1.8 V ≤ $AV_{DD}$ ≤ 3.6 V | 9.5 | | | |
| | | ADTYP = 0, 8-bit resolution Note 2 | 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V | 57.5 | | | |
| | | ADTYP = 1, 8-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | 3.3125 | | | μs |
| | | | 1.8 V ≤ $AV_{DD}$ ≤ 3.6 V | 7.875 | | | |
| | | | 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V | 54.25 | | | |
| Zero-scale error Note 3 | EZS | 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±8.0 | LSB |
| | | 10-bit resolution | 1.8 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±5.5 | |
| | | 8-bit resolution | 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±3.0 | |
| Full-scale error Note 3 | EFS | 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±8.0 | LSB |
| | | 10-bit resolution | 1.8 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±5.5 | |
| | | 8-bit resolution | 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±3.0 | |
| Integral linearity error Note 3 | ILE | 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±3.5 | LSB |
| | | 10-bit resolution | 1.8 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.5 | |
| | | 8-bit resolution | 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±1.5 | |
| Differential linearity error Note 3 | DLE | 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.5 | LSB |
| | | 10-bit resolution | 1.8 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.5 | |
| | | 8-bit resolution | 1.6 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.0 | |
| Analog input voltage | VAIN | | | 0 | | $AV_{DD}$ and $EV_{DD0}$ | V |
| | | Interanal reference voltage (2.4 V ≤ $V_{DD}$ ≤ 3.6 V, HS (high-speed main) mode) | | $V_{BGR}$ Note 4 | | | V |
| | | Temperature sensor output voltage (2.4 V ≤ $V_{DD}$ ≤ 3.6 V, HS (high-speed main) mode) | | $V_{TMPS25}$ Note 4 | | | V |

**Notes 1.** Cannot be used for lower 2 bit of ADCR register

**2.** Cannot be used for lower 4 bit of ADCR register

**3.** Excludes quantization error (±1/2 LSB).

**4.** See **29.6.2 Temperature sensor, internal reference voltage output characteristics**.

<R> **(6) When reference voltage (+) = Internal reference voltage (1.45 V) (ADREFP1 = 1, ADREFP0 = 0), reference voltage (–) = $AV_{SS}$ (ADREFM = 0), target ANI pin: ANI0 to ANI12, ANI16 to ANI30**

**($T_A$ = –40 to +85°C, 2.4 V ≤ $V_{DD}$ ≤ 3.6 V, 1.6 V ≤ $EV_{DD}$ ≤ $V_{DD}$, 1.6 V ≤ $AV_{DD}$ ≤ $V_{DD}$, $V_{SS}$ = $EV_{SS0}$ = 0 V, $AV_{SS}$ = 0 V, Reference voltage (+) = Internal reference voltage, Reference voltage (–) = $AV_{SS}$ = 0 V, HS (high-speed main) mode)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Resolution | $R_{ES}$ | | | 8 | | bit |
| Conversion time | $t_{CONV}$ | 8-bit resolution | 16 | | | μs |
| Zero-scale error[Note] | $E_{ZS}$ | 8-bit resolution | | | ±4.0 | LSB |
| Integral linearity error[Note] | ILE | 8-bit resolution | | | ±2.0 | LSB |
| Differential linearity error[Note] | DLE | 8-bit resolution | | | ±2.5 | LSB |
| Reference voltage (+) | $AV_{REF(+)}$ | = Internal reference voltage ($V_{BGR}$) | 1.38 | 1.45 | 1.5 | V |
| Analog input voltage | $V_{AIN}$ | | 0 | | $V_{BGR}$ | V |

**Note** Excludes quantization error (±1/2 LSB).

### 29.6.2 Temperature sensor, internal reference voltage output characteristics

**($T_A$ = –40 to +85°C, 2.4 V ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = 0 V, HS (high-speed main) mode)**

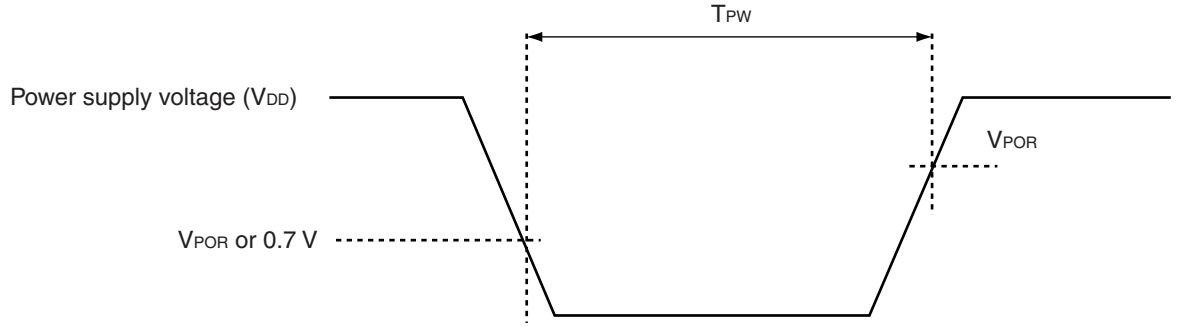
| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Temperature sensor output voltage | $V_{TMPS25}$ | Setting ADS register = 80H, $T_A$ = +25°C | | 1.05 | | V |
| Internal reference voltage | $V_{BGR}$ | Setting ADS register = 81H | 1.38 | 1.45 | 1.5 | V |
| Temperature coefficient | $F_{VTMPS}$ | Temperature sensor output voltage that depends on the temperature | | –3.6 | | mV/°C |
| Operation stabilization wait time | $t_{AMP}$ | | 10 | | | μs |

### 29.6.3 POR circuit characteristics

**($T_A$ = –40 to +85°C, $V_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Detection voltage | $V_{POR}$ | Power supply rise time | 1.47 | 1.51 | 1.55 | V |
| | $V_{PDR}$ | Power supply fall time | 1.46 | 1.50 | 1.54 | V |
| Minimum pulse width[Note] | $T_{PW}$ | | 300 | | | μs |

**Note** This is the time required for the POR circuit to execute a reset when $V_{DD}$ falls below $V_{PDR}$. When the microcontroller enters STOP mode or if the main system clock ($f_{MAIN}$) has been stopped by setting bit 0 (HIOSTOP) and bit 7 (MSTOP) of the clock operation status control register (CSC), this is the time required for the POR circuit to execute a reset before $V_{DD}$ rises to $V_{POR}$ after having fallen below 0.7 V.



### 29.6.4 LVD circuit characteristics

**LVD Detection Voltage of Reset Mode and Interrupt Mode**

**($T_A$ = −40 to +85°C, $V_{PDR} \le V_{DD} \le 3.6$ V, $V_{SS}$ = 0 V)**

| Parameter | | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Detection voltage | Supply voltage level | $V_{LVD2}$ | Power supply rise time | 3.07 | 3.13 | 3.19 | V |
| | | | Power supply fall time | 3.00 | 3.06 | 3.12 | V |
| | | $V_{LVD3}$ | Power supply rise time | 2.96 | 3.02 | 3.08 | V |
| | | | Power supply fall time | 2.90 | 2.96 | 3.02 | V |
| | | $V_{LVD4}$ | Power supply rise time | 2.86 | 2.92 | 2.97 | V |
| | | | Power supply fall time | 2.80 | 2.86 | 2.91 | V |
| | | $V_{LVD5}$ | Power supply rise time | 2.76 | 2.81 | 2.87 | V |
| | | | Power supply fall time | 2.70 | 2.75 | 2.81 | V |
| | | $V_{LVD6}$ | Power supply rise time | 2.66 | 2.71 | 2.76 | V |
| | | | Power supply fall time | 2.60 | 2.65 | 2.70 | V |
| | | $V_{LVD7}$ | Power supply rise time | 2.56 | 2.61 | 2.66 | V |
| | | | Power supply fall time | 2.50 | 2.55 | 2.60 | V |
| | | $V_{LVD8}$ | Power supply rise time | 2.45 | 2.50 | 2.55 | V |
| | | | Power supply fall time | 2.40 | 2.45 | 2.50 | V |
| | | $V_{LVD9}$ | Power supply rise time | 2.05 | 2.09 | 2.13 | V |
| | | | Power supply fall time | 2.00 | 2.04 | 2.08 | V |
| | | $V_{LVD10}$ | Power supply rise time | 1.94 | 1.98 | 2.02 | V |
| | | | Power supply fall time | 1.90 | 1.94 | 1.98 | V |
| | | $V_{LVD11}$ | Power supply rise time | 1.84 | 1.88 | 1.91 | V |
| | | | Power supply fall time | 1.80 | 1.84 | 1.87 | V |
| | | $V_{LVD12}$ | Power supply rise time | 1.74 | 1.77 | 1.81 | V |
| | | | Power supply fall time | 1.70 | 1.73 | 1.77 | V |
| | | $V_{LVD13}$ | Power supply rise time | 1.64 | 1.67 | 1.70 | V |
| | | | Power supply fall time | 1.60 | 1.63 | 1.66 | V |
| Minimum pulse width | | $t_{LW}$ | | 300 | | | μs |
| Detection delay time | | | | | | 300 | μs |

**Caution Set the detection voltage ($V_{LVD}$) to be within the operating voltage range. The operating voltage range depends on the setting of the user option byte (000C2H/010C2H). The following shows the operating voltage range.**

**HS (high-speed main) mode: $V_{DD}$ = 2.7 to 3.6 V@1 MHz to 32 MHz**
**$V_{DD}$ = 2.4 to 3.6 V@1 MHz to 16 MHz**
**LS (low-speed main) mode: $V_{DD}$ = 1.8 to 3.6 V@1 MHz to 8 MHz**
**LV (low-voltage main) mode: $V_{DD}$ = 1.6 to 3.6 V@1 MHz to 4 MHz**

**LVD Detection Voltage of Interrupt & Reset Mode**

**($T_A$ = −40 to +85°C, $V_{PDR} \le V_{DD} \le 3.6$ V, $V_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Interrupt & reset mode | $V_{LVD13}$ | VPOC2, VPOC1, VPOC0 = 0, 0, 0, falling reset voltage | | 1.60 | 1.63 | 1.66 | V |
| | $V_{LVD12}$ | LVIS1, LVIS0 = 1, 0 | Rising release reset voltage | 1.74 | 1.77 | 1.81 | V |
| | | | Falling interrupt voltage | 1.70 | 1.73 | 1.77 | V |
| | $V_{LVD11}$ | LVIS1, LVIS0 = 0, 1 | Rising release reset voltage | 1.84 | 1.88 | 1.91 | V |
| | | | Falling interrupt voltage | 1.80 | 1.84 | 1.87 | V |
| | $V_{LVD4}$ | LVIS1, LVIS0 = 0, 0 | Rising release reset voltage | 2.86 | 2.92 | 2.97 | V |
| | | | Falling interrupt voltage | 2.80 | 2.86 | 2.91 | V |
| | $V_{LVD11}$ | VPOC2, VPOC1, VPOC0 = 0, 0, 1, falling reset voltage | | 1.80 | 1.84 | 1.87 | V |
| | $V_{LVD10}$ | LVIS1, LVIS0 = 1, 0 | Rising release reset voltage | 1.94 | 1.98 | 2.02 | V |
| | | | Falling interrupt voltage | 1.90 | 1.94 | 1.98 | V |
| | $V_{LVD9}$ | LVIS1, LVIS0 = 0, 1 | Rising release reset voltage | 2.05 | 2.09 | 2.13 | V |
| | | | Falling interrupt voltage | 2.00 | 2.04 | 2.08 | V |
| | $V_{LVD2}$ | LVIS1, LVIS0 = 0, 0 | Rising release reset voltage | 3.07 | 3.13 | 3.19 | V |
| | | | Falling interrupt voltage | 3.00 | 3.06 | 3.12 | V |
| | $V_{LVD8}$ | VPOC2, VPOC1, VPOC0 = 0, 1, 0, falling reset voltage | | 2.40 | 2.45 | 2.50 | V |
| | $V_{LVD7}$ | LVIS1, LVIS0 = 1, 0 | Rising release reset voltage | 2.56 | 2.61 | 2.66 | V |
| | | | Falling interrupt voltage | 2.50 | 2.55 | 2.60 | V |
| | $V_{LVD6}$ | LVIS1, LVIS0 = 0, 1 | Rising release reset voltage | 2.66 | 2.71 | 2.76 | V |
| | | | Falling interrupt voltage | 2.60 | 2.65 | 2.70 | V |
| | $V_{LVD5}$ | VPOC2, VPOC1, VPOC0 = 0, 1, 1, falling reset voltage | | 2.70 | 2.75 | 2.81 | V |
| | $V_{LVD4}$ | LVIS1, LVIS0 = 1, 0 | Rising release reset voltage | 2.86 | 2.92 | 2.97 | V |
| | | | Falling interrupt voltage | 2.80 | 2.86 | 2.91 | V |
| | $V_{LVD3}$ | LVIS1, LVIS0 = 0, 1 | Rising release reset voltage | 2.96 | 3.02 | 3.08 | V |
| | | | Falling interrupt voltage | 2.90 | 2.96 | 3.02 | V |

**Caution Set the detection voltage ($V_{LVD}$) to be within the operating voltage range. The operating voltage range depends on the setting of the user option byte (000C2H/010C2H). The following shows the operating voltage range.**

**HS (high-speed main) mode: $V_{DD}$ = 2.7 to 3.6 V@1 MHz to 32 MHz**
**$V_{DD}$ = 2.4 to 3.6 V@1 MHz to 16 MHz**
**LS (low-speed main) mode: $V_{DD}$ = 1.8 to 3.6 V@1 MHz to 8 MHz**
**LV (low-voltage main) mode: $V_{DD}$ = 1.6 to 3.6 V@1 MHz to 4 MHz**

### 29.6.5 Supply voltage rise slope characteristics

**($T_A$ = −40 to +85°C, $V_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Supply voltage rise | $SV_{DD}$ | | | | 54 | V/ms |

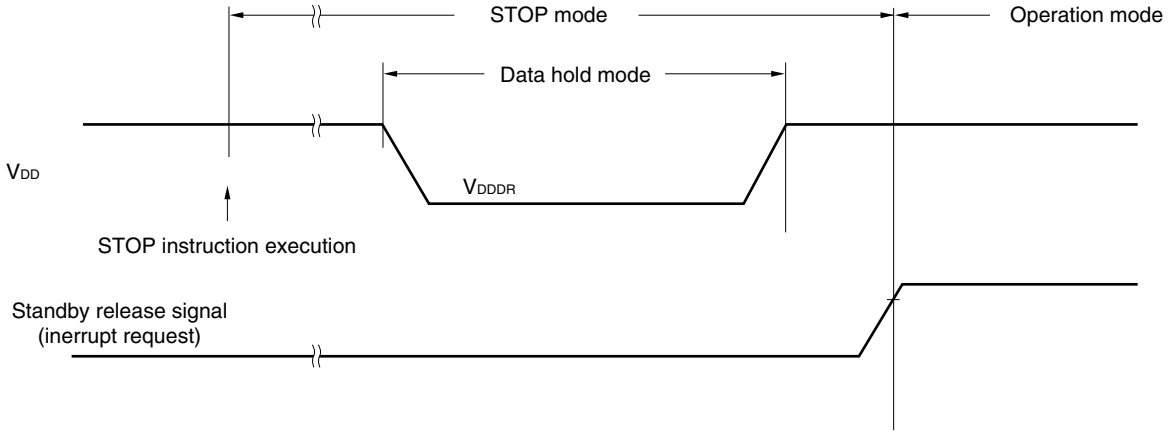
**Caution Be sure to maintain the internal reset state until $V_{DD}$ reaches the operating voltage range specified in 29.4 AC Characteristics, by using the LVD circuit or external reset pin.**

## 29.7 Data Memory STOP Mode Low Supply Voltage Data Retention Characteristics

<R> **($T_A$ = –40 to +85°C, $V_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Data retention supply voltage | $V_{DDDR}$ | | 1.46 Note | | 3.6 | V |

**Note** The value depends on the POR detection voltage. When the voltage drops, the data is retained before a POR reset is effected, but data is not retained when a POR reset is effected.




## <R> 29.8 Flash Memory Programming Characteristics

**($T_A$ = –40 to +85°C, 1.8 V ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| CPU/peripheral hardware clock frequency | $f_{CLK}$ | 1.8 V ≤ $V_{DD}$ ≤ 3.6 V | | 1 | | 32 | MHz |
| Number of code flash rewrites Notes 1, 2 | $C_{erwr}$ | Retained for 20 years | $T_A$ = 85°C Note 3 | 1,000 | | | Times |
| Number of data flash rewrites Notes 1, 2 | | Retained for 1 years | $T_A$ = 25°C Note 3 | | 1,000,000 | | |
| | | Retained for 5 years | $T_A$ = 85°C Note 3 | 100,000 | | | |
| | | Retained for 20 years | $T_A$ = 85°C Note 3 | 10,000 | | | |

**Notes 1.** 1 erase + 1 write after the erase is regarded as 1 rewrite.
The retaining years are until next rewrite after the rewrite.

**2.** When using flash memory programmer and Renesas Electronics self programming library

**3.** These are the characteristics of the flash memory and the results obtained from reliability testing by Renesas Electronics Corporation.

## 29.9 Dedicated Flash Memory Programmer Communication (UART)

**($T_A$ = –40 to +85°C, 1.8 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

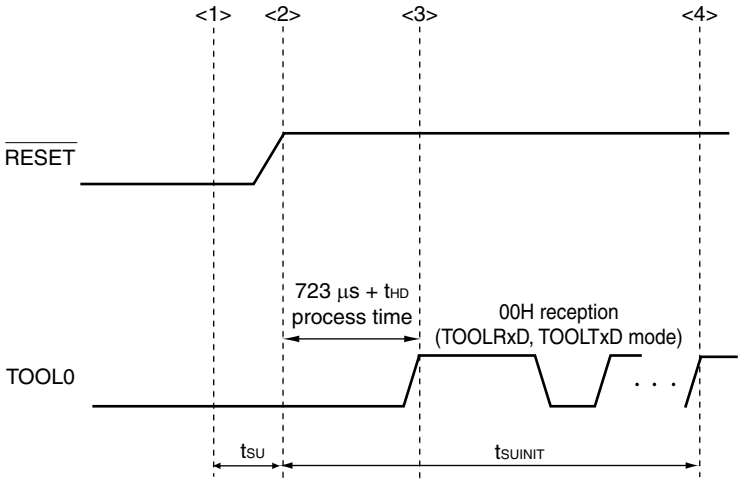
| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Transfer rate | | During flash memory programming | 115.2 k | | 1 M | bps |

## 29.10 Timing Specs for Switching Flash Memory Programming Modes

**($T_A$ = −40 to +85°C, 1.8 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| How long from when an external reset ends until the initial communication settings are specified | $t_{SUINIT}$ | POR and LVD reset must end before the external reset ends. | | | 100 | ms |
| How long from when the TOOL0 pin is placed at the low level until a external reset ends | $t_{SU}$ | POR and LVD reset must end before the external reset ends. | 10 | | | μs |
| <R> How long the TOOL0 pin must be kept at the low level after an external reset ends (except flash firmware processing time) | $t_{HD}$ | POR and LVD reset must end before the external reset ends. | 1 | | | ms |

<R>




<1> The low level is input to the TOOL0 pin.

<2> The pins reset ends (POR and LVD reset must end before the external reset ends.).

<3> The TOOL0 pin is set to the high level.

<4> Setting of the flash memory programming mode by UART reception and complete the baud rate setting.

**Remark** $t_{SUINIT}$: The segment shows that it is necessary to finish specifying the initial communication settings within 100 ms from when the resets end.

$t_{SU}$: How long from when the TOOL0 pin is placed at the low level until a external reset ends

<R> $t_{HD}$: How long to keep the TOOL0 pin at the low level from when the external resets end (except flash firmware processing time)

# CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to +105°C)

This chapter describes the following electrical specifications.

Target products G: Industrial applications $T_A$ = –40 to +105°C
R5F10EBAGNA, R5F10EBCGNA, R5F10EBDGNA, R5F10EBEGNA
R5F10EGAGFB, R5F10EGCGFB, R5F10EGDGFB, R5F10EGEGFB
R5F10EGAGNA, R5F10EGCGNA, R5F10EGDGNA, R5F10EGEGNA
R5F10ELCGFB, R5F10ELDGFB, R5F10ELEGFB

**Cautions**
1. **The RL78/G1A has an on-chip debug function, which is provided for development and evaluation. Do not use the on-chip debug function in products designated for mass production, because the guaranteed number of rewritable times of the flash memory may be exceeded when this function is used, and product reliability therefore cannot be guaranteed. Renesas Electronics is not liable for problems occurring when the on-chip debug function is used.**
2. **With products not provided with an $EV_{DD0}$ or $EV_{SS0}$ pin, replace $EV_{DD0}$ with $V_{DD}$, or replace $EV_{SS0}$ with $V_{SS}$.**
3. **The pins mounted depend on the product. See 2.1 Port Function to 2.2.1 With functions for each product.**
4. **Please contact Renesas Electronics sales office for derating of operation under $T_A$ = +85°C to +105°C. Derating is the systematic reduction of load for the sake of improved reliability.**

**Remark** When RL78/G1A is used in the range of $T_A$ = –40 to +85°C, see **CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C)**.

## 30.1 Absolute Maximum Ratings

**Absolute Maximum Ratings ($T_A$ = 25°C) (1/2)**

| Parameter | Symbols | Conditions | Ratings | Unit |
|---|---|---|---|---|
| Supply voltage | $V_{DD}$ | | –0.5 to +6.5 | V |
| | $EV_{DD0}$ | | –0.5 to +6.5 | V |
| | $AV_{DD}$ | | –0.5 to +4.6 | V |
| | $AV_{REFP}$ | | –0.3 to $AV_{DD}$ +0.3 Note 3 | V |
| | $EV_{SS0}$ | | –0.5 to +0.3 | V |
| | $AV_{SS}$ | | –0.5 to +0.3 | V |
| | $AV_{REFM}$ | | –0.3 to $AV_{DD}$ +0.3 Note 3 and $AV_{REFM} \leq AV_{REFP}$ | V |
| REGC pin input voltage | $V_{IREGC}$ | REGC | –0.3 to +2.8 and –0.3 to $V_{DD}$ +0.3 Note 1 | V |
| Input voltage | $V_{I1}$ | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P70 to P77, P120, P140, P141 | –0.3 to $EV_{DD0}$ +0.3 and –0.3 to $V_{DD}$ +0.3 Note 2 | V |
| | $V_{I2}$ | P60 to P63 (N-ch open-drain) | –0.3 to +6.5 | V |
| | $V_{I3}$ | P121 to P124, P137, EXCLK, EXCLKS, $\overline{\text{RESET}}$ | –0.3 to $V_{DD}$ +0.3 Note 2 | V |
| | $V_{I4}$ | P20 to P27, P150 to P154 | –0.3 to $AV_{DD}$ +0.3 Note 2 | V |
| Output voltage | $V_{O1}$ | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P60 to P63, P70 to P77, P120, P130, P140, P141 | –0.3 to $EV_{DD0}$ +0.3 Note 2 | V |
| | $V_{O2}$ | P20 to P27, P150 to P154 | –0.3 to $AV_{DD}$ +0.3 Note 2 | V |
| Analog input voltage | $V_{AI1}$ | ANI16 to ANI30 | –0.3 to $EV_{DD0}$ +0.3 and –0.3 to $AV_{REF(+)}$ +0.3 Notes 2, 4 | V |
| | $V_{AI2}$ | ANI0 to ANI12 | –0.3 to $AV_{DD}$ +0.3 and –0.3 to $AV_{REF(+)}$ +0.3 Notes 2, 4 | V |

**Notes 1.** Connect the REGC pin to $V_{SS}$ via a capacitor (0.47 to 1 $\mu$F). This value regulates the absolute maximum rating of the REGC pin. Do not use this pin with voltage applied to it.

**2.** Must be 6.5 V or lower.

**3.** Must be 4.6 V or lower.

**4.** Do not exceed $AV_{REF(+)}$ + 0.3 V in case of A/D conversion target pin.

**Caution Product quality may suffer if the absolute maximum rating is exceeded even momentarily for any parameter. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions that ensure that the absolute maximum ratings are not exceeded.**

**Remarks 1.** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

**2.** $AV_{REF(+)}$: + side reference voltage of the A/D converter.

**3.** $V_{SS}$: Reference voltage

**Absolute Maximum Ratings ($T_A$ = 25°C) (2/2)**

| Parameter | Symbols | Conditions | | Ratings | Unit |
|---|---|---|---|---|---|
| Output current, high | $I_{OH1}$ | Per pin | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P70 to P77, P120, P130, P140, P141 | −40 | mA |
| | | Total of all pins −170 mA | P00 to P04, P40 to P43, P120, P130, P140, P141 | −70 | mA |
| | | | P05, P06, P10 to P16, P30, P31, P50, P51, P70 to P77, | −100 | mA |
| | $I_{OH2}$ | Per pin | P20 to P27, P150 to P154 | −0.1 | mA |
| | | Total of all pins | | −1.3 | mA |
| Output current, low | $I_{OL1}$ | Per pin | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P60 to P63, P70 to P77, P120, P130, P140, P141 | 40 | mA |
| | | Total of all pins 170 mA | P00 to P04, P40 to P43, P120, P130, P140, P141 | 70 | mA |
| | | | P05, P06, P10 to P16, P30, P31, P50, P51, P60 to P63, P70 to P77 | 100 | mA |
| | $I_{OL2}$ | Per pin | P20 to P27, P150 to P154 | 0.4 | mA |
| | | Total of all pins | | 6.4 | mA |
| Operating ambient temperature | $T_A$ | In normal operation mode | | −40 to +105 | °C |
| | | In flash memory programming mode | | | |
| Storage temperature | $T_{stg}$ | | | −65 to +150 | °C |

**Caution Product quality may suffer if the absolute maximum rating is exceeded even momentarily for any parameter. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions that ensure that the absolute maximum ratings are not exceeded.**

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

## 30.2 Oscillator Characteristics

### 30.2.1 X1, XT1 oscillator characteristics

**($T_A$ = −40 to +105°C, 2.4 V ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = 0 V)**

| Parameter | Resonator | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| X1 clock oscillation frequency ($f_X$)Note | Ceramic resonator/crystal resonator | 2.7 V ≤ $V_{DD}$ ≤ 3.6 V | 1.0 | | 20.0 | MHz |
| | | 2.4 V ≤ $V_{DD}$ < 2.7 V | 1.0 | | 16.0 | |
| XT1 clock oscillation frequency ($f_X$)Note | Crystal resonator | | 32 | 32.768 | 35 | kHz |

**Note** Indicates only permissible oscillator frequency ranges. See AC Characteristics for instruction execution time. Request evaluation by the manufacturer of the oscillator circuit mounted on a board to check the oscillator characteristics.

<R> **Caution Since the CPU is started by the high-speed on-chip oscillator clock after a reset release, check the X1 clock oscillation stabilization time using the oscillation stabilization time counter status register (OSTC) by the user. Determine the oscillation stabilization time of the OSTC register and the oscillation stabilization time select register (OSTS) after sufficiently evaluating the oscillation stabilization time with the resonator to be used.**

<R> **Remark** When using the X1 oscillator and XT1 oscillator, see **5.4 System Clock Oscillator**.

### 30.2.2 On-chip oscillator characteristics

**($T_A$ = −40 to +105°C, 2.4 V ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = 0 V)**

| Oscillators | Parameters | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| High-speed on-chip oscillator oscillation frequencyNotes 1, 2 | $f_{IH}$ | | | 1 | | 32 | MHz |
| High-speed on-chip oscillator oscillation frequency accuracy | | +85 to +105 °C | 2.4 V ≤ $V_{DD}$ ≤ 3.6 V | −2 | | +2 | % |
| | | −20 to +85 °C | 2.4 V ≤ $V_{DD}$ ≤ 3.6 V | −1 | | +1 | % |
| | | −40 to −20 °C | 2.4 V ≤ $V_{DD}$ ≤ 3.6 V | −1.5 | | +1.5 | % |
| Low-speed on-chip oscillator oscillation frequency | $f_{IL}$ | | | | 15 | | kHz |
| Low-speed on-chip oscillator oscillation frequency accuracy | | | | −15 | | +15 | % |

**Notes 1.** High-speed on-chip oscillator frequency is selected by bits 0 to 3 of option byte (000C2H/010C2H) and bits 0 to 2 of HOCODIV register.

**2.** This indicates the oscillator characteristics only. See AC Characteristics for instruction execution time.

## 30.3 DC Characteristics

### 30.3.1 Pin characteristics

<R> **($T_A$ = −40 to +105°C, 2.4 V ≤ $AV_{DD}$ ≤ $V_{DD}$ ≤ 3.6 V, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)** **(1/5)**

| Items | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Output current, high[Note 1] | $I_{OH1}$ | Per pin for P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P70 to P77, P120, P130, P140, P141 | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | −3.0[Note 2] | mA |
| | | Total of P00 to P04, P40 to P43, P120, P130, P140, P141 (When duty ≤ 70%[Note 3]) | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | −10.0 | mA |
| | | | 2.4 V ≤ $EV_{DD0}$ < 2.7 V | | | −5.0 | mA |
| | | Total of P05, P06, P10 to P16, P30, P31, P50, P51, P70 to P77, (When duty ≤ 70%[Note 3]) | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | −19.0 | mA |
| | | | 2.4 V ≤ $EV_{DD0}$ < 2.7 V | | | −10.0 | mA |
| | | Total of all pins (When duty ≤ 70%[Note 3]) | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | −29.0 | mA |
| | $I_{OH2}$ | Per pin for P20 to P27, P150 to P154 | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | −0.1[Note 2] | mA |
| | | Total of all pins (When duty ≤ 70%[Note 3]) | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | −1.3 | mA |

**Notes 1.** Value of current at which the device operation is guaranteed even if the current flows from the $EV_{DD0}$, $V_{DD}$ pins to an output pin.

**2.** However, do not exceed the total current value.

**3.** Specification under conditions where the duty factor ≤ 70%.
The output current value that has changed to the duty factor > 70% the duty ratio can be calculated with the following expression (when changing the duty factor from 70% to n%).

- Total output current of pins = ($I_{OH}$ × 0.7)/(n × 0.01)

<Example> Where n = 80% and $I_{OH}$ = −10.0 mA
Total output current of pins = (−10.0 × 0.7)/(80 × 0.01) ≅ −8.7 mA

However, the current that is allowed to flow into one pin does not vary depending on the duty factor. A current higher than the absolute maximum rating must not flow into one pin.

**Caution P00, P02 to P04, P10 to P15, P43, P50, P71, and P74 do not output high level in N-ch open-drain mode.**

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

<R> **($T_A$ = −40 to +105°C, 2.4 V ≤ $AV_{DD}$ ≤ $V_{DD}$ ≤ 3.6 V, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)** **(2/5)**

| Items | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Output current, low[Note 1] | $I_{OL1}$ | Per pin for P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P70 to P77, P120, P130, P140, P141 | | | | 8.5[Note 2] | mA |
| | | Per pin for P60 to P63 | | | | 15.0[Note 2] | mA |
| | | Total of P00 to P04, P40 to P43, P120, P130, P140, P141 (When duty ≤ 70%[Note 3]) | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | 15.0 | mA |
| | | | 2.4 V ≤ $EV_{DD0}$ < 2.7 V | | | 9.0 | mA |
| | | Total of P05, P06, P10 to P16, P30, P31, P50, P51, P60 to P63, P70 to P77 (When duty ≤ 70%[Note 3]) | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | 35.0 | mA |
| | | | 2.4 V ≤ $EV_{DD0}$ < 2.7 V | | | 20.0 | mA |
| | | Total of all pins (When duty ≤ 70%[Note 3]) | | | | 50.0 | mA |
| | $I_{OL2}$ | Per pin for P20 to P27, P150 to P154 | | | | 0.4[Note 2] | mA |
| | | Total of all pins (When duty ≤ 70%[Note 3]) | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | 5.2 | mA |

**Notes 1.** Value of current at which the device operation is guaranteed even if the current flows from an output pin to the $EV_{SS0}$ and $V_{SS}$ pin.

**2.** However, do not exceed the total current value.

**3.** Specification under conditions where the duty factor ≤ 70%.
The output current value that has changed to the dury factor > 70% the duty ratio can can be calculated with the following expression (when changing the duty factor from 70% to n%).

- Total output current of pins = ($I_{OL}$ × 0.7)/(n × 0.01)

<Example> Where n = 80% and $I_{OL}$ = 10.0 mA
Total output current of pins = (10.0 × 0.7)/(80 × 0.01) ≅ 8.7 mA

However, the current that is allowed to flow into one pin does not vary depending on the duty factor. A current higher than the absolute maximum rating must not flow into one pin.

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

**($T_A$ = −40 to +105°C, 2.4 V ≤ $AV_{DD}$ ≤ $V_{DD}$ ≤ 3.6 V, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)** **(3/5)**

| Items | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Input voltage, high | $V_{IH1}$ | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P70 to P77, P120, P140, P141 | Normal input buffer | 0.8$EV_{DD0}$ | | $EV_{DD0}$ | V |
| | $V_{IH2}$ | P01, P03, P04, P10, P11, P13 to P16, P43 | TTL input buffer 3.3 V ≤ $EV_{DD0}$ ≤ 3.6 V | 2.0 | | $EV_{DD0}$ | V |
| | | | TTL input buffer 2.4 V ≤ $EV_{DD0}$ < 3.3 V | 1.5 | | $EV_{DD0}$ | V |
| | $V_{IH3}$ | P20 to P27, P150 to P154 | | 0.7$AV_{DD}$ | | $AV_{DD}$ | V |
| | $V_{IH4}$ | P60 to P63 | | 0.7$EV_{DD0}$ | | 6.0 | V |
| | $V_{IH5}$ | P121 to P124, P137, EXCLK, EXCLKS, $\overline{\text{RESET}}$ | | 0.8$V_{DD}$ | | $V_{DD}$ | V |
| Input voltage, low | $V_{IL1}$ | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P70 to P77, P120, P140, P141 | Normal input buffer | 0 | | 0.2$EV_{DD0}$ | V |
| | $V_{IL2}$ | P01, P03, P04, P10, P11, P13 to P16, P43 | TTL input buffer 3.3 V ≤ $EV_{DD0}$ ≤ 3.6 V | 0 | | 0.5 | V |
| | | | TTL input buffer 2.4 V ≤ $EV_{DD0}$ < 3.3 V | 0 | | 0.32 | V |
| | $V_{IL3}$ | P20 to P27, P150 to P154 | | 0 | | 0.3$AV_{DD}$ | V |
| | $V_{IL4}$ | P60 to P63 | | 0 | | 0.3$EV_{DD0}$ | V |
| | $V_{IL5}$ | P121 to P124, P137, EXCLK, EXCLKS, $\overline{\text{RESET}}$ | | 0 | | 0.2$V_{DD}$ | V |

**Caution The maximum value of $V_{IH}$ of pins P00, P02 to P04, P10 to P15, P43, P50, P71, and P74 is $EV_{DD0}$, even in the N-ch open-drain mode.**

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

**($T_A$ = −40 to +105°C, 2.4 V ≤ $AV_{DD}$ ≤ $V_{DD}$ ≤ 3.6 V, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)** **(4/5)**

| Items | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Output voltage, high | $V_{OH1}$ | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P70 to P77, P120, P130, P140, P141 | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, $I_{OH1}$ = −2.0 mA | $EV_{DD0}$ − 0.6 | | | V |
| | | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V, $I_{OH1}$ = −1.5 mA | $EV_{DD0}$ − 0.5 | | | V |
| | $V_{OH2}$ | P20 to P27, P150 to P154 | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V, $I_{OH2}$ = −100 μA | $AV_{DD}$ − 0.5 | | | V |
| Output voltage, low | $V_{OL1}$ | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P70 to P77, P120, P130, P140, P141 | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, $I_{OL1}$ = 3.0 mA | | | 0.6 | V |
| | | | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, $I_{OL1}$ = 1.5 mA | | | 0.4 | V |
| | | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V, $I_{OL1}$ = 0.6 mA | | | 0.4 | V |
| | $V_{OL2}$ | P20 to P27, P150 to P154 | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V, $I_{OL2}$ = 400 μA | | | 0.4 | V |
| | $V_{OL3}$ | P60 to P63 | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, $I_{OL3}$ = 3.0 mA | | | 0.4 | V |
| | | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V, $I_{OL3}$ = 2.0 mA | | | 0.4 | V |

**Caution P00, P02 to P04, P10 to P15, P43, P50, P71, and P74 do not output high level in N-ch open-drain mode.**

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

**($T_A$ = −40 to +105°C, 2.4 V ≤ $AV_{DD}$ ≤ $V_{DD}$ ≤ 3.6 V, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)** **(5/5)**

| Items | Symbol | Conditions | | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|---|
| Input leakage current, high | $I_{LIH1}$ | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P60 to P63, P70 to P77, P120, P140, P141 | $V_I$ = $EV_{DD0}$ | | | | 1 | μA |
| | $I_{LIH2}$ | P137, $\overline{\text{RESET}}$ | $V_I$ = $V_{DD}$ | | | | 1 | μA |
| | $I_{LIH3}$ | P121 to P124 (X1, X2, XT1, XT2, EXCLK, EXCLKS) | $V_I$ = $V_{DD}$ | In input port or external clock input | | | 1 | μA |
| | | | | In resonator connection | | | 10 | μA |
| | $I_{LIH4}$ | P20 to P27, P150 to P154 | $V_I$ = $AV_{DD}$ | | | | 1 | μA |
| Input leakage current, low | $I_{LIL1}$ | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P60 to P67, P70 to P77, P120, P140, P141 | $V_I$ = $EV_{SS0}$ | | | | −1 | μA |
| | $I_{LIL2}$ | P137, $\overline{\text{RESET}}$ | $V_I$ = $V_{SS}$ | | | | −1 | μA |
| | $I_{LIL3}$ | P121 to P124 (X1, X2, XT1, XT2, EXCLK, EXCLKS) | $V_I$ = $V_{SS}$ | In input port or external clock input | | | −1 | μA |
| | | | | In resonator connection | | | −10 | μA |
| | $I_{LIL4}$ | P20 to P27, P150 to P154 | $V_I$ = $AV_{SS}$ | | | | −1 | μA |
| On-chip pull-up resistance | $R_U$ | P00 to P06, P10 to P16, P30, P31, P40 to P43, P50, P51, P70 to P77, P120, P140, P141 | $V_I$ = $EV_{SS0}$, In input port | | 10 | 20 | 100 | kΩ |

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

### 30.3.2 Supply current characteristics

**($T_A$ = −40 to +105°C, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)** **(1/3)**

| Parameter | Symbol | Conditions | | | | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|---|---|---|
| Supply current | $I_{DD1}$ Note 1 | Operating mode | HS (high-speed main) mode Note 5 | $f_{IH}$ = 32 MHz Note 3 | Basic operation | $V_{DD}$ = 3.0 V | | 2.1 | | mA |
| | | | | | Normal operation | $V_{DD}$ = 3.0 V | | 4.6 | 7.5 | mA |
| | | | | $f_{IH}$ = 24 MHz Note 3 | Normal operation | $V_{DD}$ = 3.0 V | | 3.7 | 5.8 | mA |
| | | | | $f_{IH}$ = 16 MHz Note 3 | Normal operation | $V_{DD}$ = 3.0 V | | 2.7 | 4.2 | mA |
| | | | HS (high-speed main) mode Note 5 | $f_{MX}$ = 20 MHz Note 2, $V_{DD}$ = 3.0 V | Normal operation | Square wave input | | 3.0 | 4.9 | mA |
| | | | | | | Resonator connection | | 3.2 | 5.0 | |
| | | | | $f_{MX}$ = 10 MHz Note 2, $V_{DD}$ = 3.0 V | Normal operation | Square wave input | | 1.9 | 2.9 | mA |
| | | | | | | Resonator connection | | 1.9 | 2.9 | |
| | | | Subsystem clock mode | $f_{SUB}$ = 32.768 kHz Note 4 $T_A$ = −40°C | Normal operation | Square wave input | | 4.1 | 4.9 | μA |
| | | | | | | Resonator connection | | 4.2 | 5.0 | |
| | | | | $f_{SUB}$ = 32.768 kHz Note 4 $T_A$ = +25°C | Normal operation | Square wave input | | 4.2 | 4.9 | μA |
| | | | | | | Resonator connection | | 4.3 | 5.0 | |
| | | | | $f_{SUB}$ = 32.768 kHz Note 4 $T_A$ = +50°C | Normal operation | Square wave input | | 4.3 | 5.5 | μA |
| | | | | | | Resonator connection | | 4.4 | 5.6 | |
| | | | | $f_{SUB}$ = 32.768 kHz Note 4 $T_A$ = +70°C | Normal operation | Square wave input | | 4.5 | 6.3 | μA |
| | | | | | | Resonator connection | | 4.6 | 6.4 | |
| | | | | $f_{SUB}$ = 32.768 kHz Note 4 $T_A$ = +85°C | Normal operation | Square wave input | | 4.8 | 7.7 | μA |
| | | | | | | Resonator connection | | 4.9 | 7.8 | |
| | | | | $f_{SUB}$ = 32.768 kHz Note 4 $T_A$ = +105°C | Normal operation | Square wave input | | 6.9 | 19.7 | μA |
| | | | | | | Resonator connection | | 7.0 | 19.8 | |

(**Notes** and **Remarks** are listed on the next page.)

**Notes 1.** Total current flowing into $V_{DD}$ and $EV_{DD0}$, including the input leakage current flowing when the level of the input pin is fixed to $V_{DD}$, $EV_{DD0}$ or $V_{SS}$, $EV_{SS0}$. The values below the MAX. column include the peripheral operation current. However, not including the current flowing into the A/D converter, LVD circuit, I/O port, on-chip pull-up/pull-down resistors, and data flash rewriting.

**2.** When high-speed on-chip oscillator and subsystem clock are stopped.

**3.** When high-speed system clock and subsystem clock are stopped.

**4.** When high-speed on-chip oscillator and high-speed system clock are stopped. When setting ultra-low power consumption oscillation (AMPHS1 = 1). Not including the current flowing into the RTC, 12-bit interval timer and watchdog timer

**5.** Relationship between operation voltage width, operation frequency of CPU and operation mode is as below.
HS (high-speed main) mode: $V_{DD}$ = 2.7 V to 3.6 V@1 MHz to 32 MHz
$V_{DD}$ = 2.4 V to 3.6 V@1 MHz to 16 MHz

**Remarks 1.** $f_{MX}$: High-speed system clock frequency (X1 clock oscillation frequency or external main system clock frequency)

**2.** $f_{IH}$: High-speed on-chip oscillator clock frequency

**3.** $f_{SUB}$: Subsystem clock frequency (XT1 clock oscillation frequency)

**4.** Except subsystem clock operation, temperature condition of the TYP. value is $T_A$ = 25°C

**($T_A$ = −40 to +105°C, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)** **(2/3)**

| Parameter | Symbol | Conditions | | | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|---|---|
| Supply current[Note 1] | $I_{DD2}$[Note 2] | HALT mode | HS (high-speed main) mode[Note 7] | $f_{IH}$ = 32 MHz[Note 4] | $V_{DD}$ = 3.0 V | | 0.54 | 2.90 | mA |
| | | | | $f_{IH}$ = 24 MHz[Note 4] | $V_{DD}$ = 3.0 V | | 0.44 | 2.30 | mA |
| | | | | $f_{IH}$ = 16 MHz[Note 4] | $V_{DD}$ = 3.0 V | | 0.40 | 1.70 | mA |
| | | | HS (high-speed main) mode[Note 7] | $f_{MX}$ = 20 MHz[Note 3], $V_{DD}$ = 3.0 V | Square wave input | | 0.28 | 1.90 | mA |
| | | | | | Resonator connection | | 0.45 | 2.00 | |
| | | | | $f_{MX}$ = 10 MHz[Note 3], $V_{DD}$ = 3.0 V | Square wave input | | 0.19 | 1.02 | mA |
| | | | | | Resonator connection | | 0.26 | 1.10 | |
| | | | Subsystem clock mode | $f_{SUB}$ = 32.768 kHz[Note 5] $T_A$ = −40°C | Square wave input | | 0.25 | 0.57 | μA |
| | | | | | Resonator connection | | 0.44 | 0.76 | |
| | | | | $f_{SUB}$ = 32.768 kHz[Note 5] $T_A$ = +25°C | Square wave input | | 0.30 | 0.57 | μA |
| | | | | | Resonator connection | | 0.49 | 0.76 | |
| | | | | $f_{SUB}$ = 32.768 kHz[Note 5] $T_A$ = +50°C | Square wave input | | 0.38 | 1.17 | μA |
| | | | | | Resonator connection | | 0.57 | 1.36 | |
| | | | | $f_{SUB}$ = 32.768 kHz[Note 5] $T_A$ = +70°C | Square wave input | | 0.52 | 1.97 | μA |
| | | | | | Resonator connection | | 0.71 | 2.16 | |
| | | | | $f_{SUB}$ = 32.768 kHz[Note 5] $T_A$ = +85°C | Square wave input | | 0.97 | 3.37 | μA |
| | | | | | Resonator connection | | 1.16 | 3.56 | |
| | | | | $f_{SUB}$ = 32.768 kHz[Note 5] $T_A$ = +105°C | Square wave input | | 3.01 | 15.37 | μA |
| | | | | | Resonator connection | | 3.20 | 15.56 | |
| | $I_{DD3}$[Note 6] | STOP mode[Note 8] | $T_A$ = −40°C | | | | 0.16 | 0.50 | μA |
| | | | $T_A$ = +25°C | | | | 0.23 | 0.50 | |
| | | | $T_A$ = +50°C | | | | 0.34 | 1.10 | |
| | | | $T_A$ = +70°C | | | | 0.46 | 1.90 | |
| | | | $T_A$ = +85°C | | | | 0.75 | 3.30 | |
| | | | $T_A$ = +105°C | | | | 2.94 | 15.30 | |

(**Notes** and **Remarks** are listed on the next page.)

**Notes 1.** Total current flowing into $V_{DD}$ and $EV_{DD0}$, including the input leakage current flowing when the level of the input pin is fixed to $V_{DD}$, $EV_{DD0}$ or $V_{SS}$, $EV_{SS0}$. The values below the MAX. column include the peripheral operation current. However, not including the current flowing into the A/D converter, LVD circuit, I/O port, on-chip pull-up/pull-down resistors, and data flash rewriting.

**2.** During HALT instruction execution by flash memory.

**3.** When high-speed on-chip oscillator and subsystem clock are stopped.

**4.** When high-speed system clock and subsystem clock are stopped.

**5.** When high-speed on-chip oscillator and high-speed system clock are stopped. When RTCLPC = 1 and setting ultra-low current consumption (AMPHS1 = 1). Including the current flowing into the RTC. However, not including the current flowing into the 12-bit interval timer, and watchdog timer.

**6.** When subsystem clock is stopped. Not including the current flowing into the RTC, 12-bit interval timer, watchdog timer.

**7.** Relationship between operation voltage width, operation frequency of CPU and operation mode is as below.

HS (high-speed main) mode: 2.7 V ≤ $V_{DD}$ ≤ 3.6 V@1 MHz to 32 MHz

2.4 V ≤ $V_{DD}$ ≤ 3.6 V@1 MHz to 16 MHz

**8.** Regarding the value for current to operate the subsystem clock in STOP mode, refer to that in HALT mode.

**Remarks 1.** $f_{MX}$: High-speed system clock frequency (X1 clock oscillation frequency or external main system clock frequency)

**2.** $f_{IH}$: High-speed on-chip oscillator clock frequency

**3.** $f_{SUB}$: Subsystem clock frequency (XT1 clock oscillation frequency)

**4.** Except subsystem clock operation and STOP mode, temperature condition of the TYP. value is $T_A$ = 25°C

**($T_A$ = −40 to +105°C, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)** **(3/3)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Low-speed on-chip oscillator operating current | $I_{FIL}$ Note 1 | | | | 0.20 | | μA |
| RTC operating current | $I_{RTC}$ Notes 1, 2, 3 | | | | 0.02 | | μA |
| 12-bit interval timer operating current | $I_{IT}$ Notes 1, 2, 4 | | | | 0.02 | | μA |
| Watchdog timer operating current | $I_{WDT}$ Notes 1, 2, 5 | $f_{IL}$ = 15 kHz | | | 0.22 | | μA |
| A/D converter operating current | $I_{ADC}$ Notes 6, 7 | $AV_{DD}$ = 3.0 V, When conversion at maximum speed | | | 420 | 720 | μA |
| $AV_{REF(+)}$ current | $I_{AVREF}$ Note 8 | $AV_{DD}$ = 3.0 V, ADREFP1 = 0, ADREFP0 = 0 Note 7 | | | 14.0 | 25.0 | μA |
| | | $AV_{REFP}$ = 3.0 V, ADREFP1 = 0, ADREFP0 = 1 Note 10 | | | 14.0 | 25.0 | μA |
| | | ADREFP1 = 1, ADREFP0 = 0 Note 1 | | | 14.0 | 25.0 | μA |
| A/D converter reference voltage current | $I_{ADREF}$ Notes 1, 9 | $V_{DD}$ = 3.0 V | | | 75.0 | | μA |
| Temperature sensor operating current | $I_{TMPS}$ Note 1 | $V_{DD}$ = 3.0 V | | | 75.0 | | μA |
| LVD operating current | $I_{LVD}$ Notes 1, 11 | | | | 0.08 | | μA |
| BGO operating current | $I_{BGO}$ Notes 1, 12 | | | | 2.5 | 12.2 | mA |
| Self-programming operating current | $I_{FSP}$ Notes 1, 13 | | | | 2.5 | 12.2 | mA |
| SNOOZE operating current | $I_{SNOZ}$ | A/D converter operation ($AV_{DD}$ = 3.0 V) | The mode is performed Notes 1, 14 | | 0.50 | 1.10 | mA |
| | | | During A/D conversion Note 1 | | 0.60 | 1.34 | mA |
| | | | During A/D conversion Note 7 | | 420 | 720 | μA |
| | | CSI/UART operation Note 1 | | | 0.70 | 1.54 | mA |

(**Notes** and **Remarks** are listed on the next page.)

<R> **Notes 1.** Current flowing to $V_{DD}$.

**2.** When high-speed on-chip oscillator and high-speed system clock are stopped.

**3.** Current flowing only to the real-time clock (RTC) (excluding the operating current of the low-speed on-chip ocsillator and the XT1 oscillator). The supply current of the RL78 microcontrollers is the sum of the values of either $I_{DD1}$ or $I_{DD2}$, and $I_{RTC}$, when the real-time clock operates in operation mode or HALT mode. When the low-speed on-chip oscillator is selected, $I_{FIL}$ should be added. $I_{DD2}$ subsystem clock operation includes the operational current of the real-time clock.

**4.** Current flowing only to the 12-bit interval timer (excluding the operating current of the low-speed on-chip ocsillator and the XT1 oscillator). The supply current of the RL78 microcontrollers is the sum of the values of either $I_{DD1}$ or $I_{DD2}$, and $I_{IT}$, when the 12-bit interval timer operates in operation mode or HALT mode. When the low-speed on-chip oscillator is selected, $I_{FIL}$ should be added.

**5.** Current flowing only to the watchdog timer (including the operating current of the low-speed on-chip oscillator). The supply current of the RL78 microcontrollers is the sum of $I_{DD1}$, $I_{DD2}$, or $I_{DD3}$ and $I_{WDT}$ when the watchdog timer is in operation.

**6.** Current flowing only to the A/D converter. The supply current of the RL78 microcontrollers is the sum of $I_{DD1}$ or $I_{DD2}$ and $I_{ADC}$, $I_{AVREF}$, $I_{ADREF}$ when the A/D converter operates in an operation mode or the HALT mode.

**7.** Current flowing to the $AV_{DD}$.

**8.** Current flowing from the reference voltage source of A/D converter.

**9.** Operation current flowing to the internal reference voltage.

**10.** Current flowing to the $AV_{REFP}$.

**11.** Current flowing only to the LVD circuit. The supply current of the RL78 microcontrollers is the sum of $I_{DD1}$, $I_{DD2}$ or $I_{DD3}$ and $I_{LVD}$ when the LVD circuit is in operation.

**12.** Current flowing only during data flash rewrite.

**13.** Current flowing only during self programming.

**14.** For shift time to the SNOOZE mode, see **18.3.3 SNOOZE mode.**

**Remarks 1.** $f_{IL}$: Low-speed on-chip oscillator clock frequency

**2.** $f_{SUB}$: Subsystem clock frequency (XT1 clock oscillation frequency)

**3.** $f_{CLK}$: CPU/peripheral hardware clock frequency

**4.** Temperature condition of the TYP. value is $T_A$ = 25°C

## 30.4 AC Characteristics

**($T_A$ = −40 to +105°C, $AV_{DD} \le V_{DD} \le 3.6$ V, 2.4 V ≤ $EV_{DD0} \le V_{DD} \le 3.6$ V, $V_{SS} = EV_{SS0}$ = 0 V)**

| Items | Symbol | Conditions | | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|---|
| Instruction cycle (minimum instruction execution time) | $T_{CY}$ | Main system clock ($f_{MAIN}$) operation | HS (high-speed main) mode | 2.7 V ≤ $V_{DD}$ ≤ 3.6 V | 0.03125 | | 1 | μs |
| | | | | 2.4 V ≤ $V_{DD}$ < 2.7 V | 0.0625 | | 1 | μs |
| | | Subsystem clock ($f_{SUB}$) operation | | 2.4 V ≤ $V_{DD}$ ≤ 3.6 V | 28.5 | 30.5 | 31.3 | μs |
| | | In the self programming mode | HS (high-speed main) mode | 2.7 V ≤ $V_{DD}$ ≤ 3.6 V | 0.03125 | | 1 | μs |
| | | | | 2.4 V ≤ $V_{DD}$ < 2.7 V | 0.0625 | | 1 | μs |
| External system clock frequency | $f_{EX}$ | 2.7 V ≤ $V_{DD}$ ≤ 3.6 V | | | 1.0 | | 20.0 | MHz |
| | | 2.4 V ≤ $V_{DD}$ < 2.7 V | | | 1.0 | | 16.0 | MHz |
| | $f_{EXS}$ | | | | 32 | | 35 | kHz |
| External system clock input high-level width, low-level width | $t_{EXH}$, $t_{EXL}$ | 2.7 V ≤ $V_{DD}$ ≤ 3.6 V | | | 24 | | | ns |
| | | 2.4 V ≤ $V_{DD}$ < 2.7 V | | | 30 | | | ns |
| | $t_{EXHS}$, $t_{EXLS}$ | | | | 13.7 | | | μs |
| TI00, TI01, TI03 to TI07 input high-level width, low-level width | $t_{TIH}$, $t_{TIL}$ | | | | 1/$f_{MCK}$+10 | | | ns[Note] |
| TO00, TO01, TO03 to TO07 output frequency | $f_{TO}$ | HS (high-speed main) mode | | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | 8 | MHz |
| | | | | 2.4 V ≤ $EV_{DD0}$ < 2.7 V | | | 4 | MHz |
| PCLBUZ0, PCLBUZ1 output frequency | $f_{PCL}$ | HS (high-speed main) mode | | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | 8 | MHz |
| | | | | 2.4 V ≤ $EV_{DD0}$ < 2.7 V | | | 4 | MHz |
| Interrupt input high-level width, low-level width | $t_{INTH}$, $t_{INTL}$ | INTP0 | | 2.4 V ≤ $V_{DD}$ ≤ 3.6 V | 1 | | | μs |
| | | INTP1 to INTP11 | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V | 1 | | | μs |
| Key interrupt input high-level width, low-level width | $t_{KR}$ | KR0 to KR9 | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.4 V ≤ $AV_{DD0}$ ≤ 3.6 V | 250 | | | ns |
| $\overline{\text{RESET}}$ low-level width | $t_{RSL}$ | | | | 10 | | | μs |

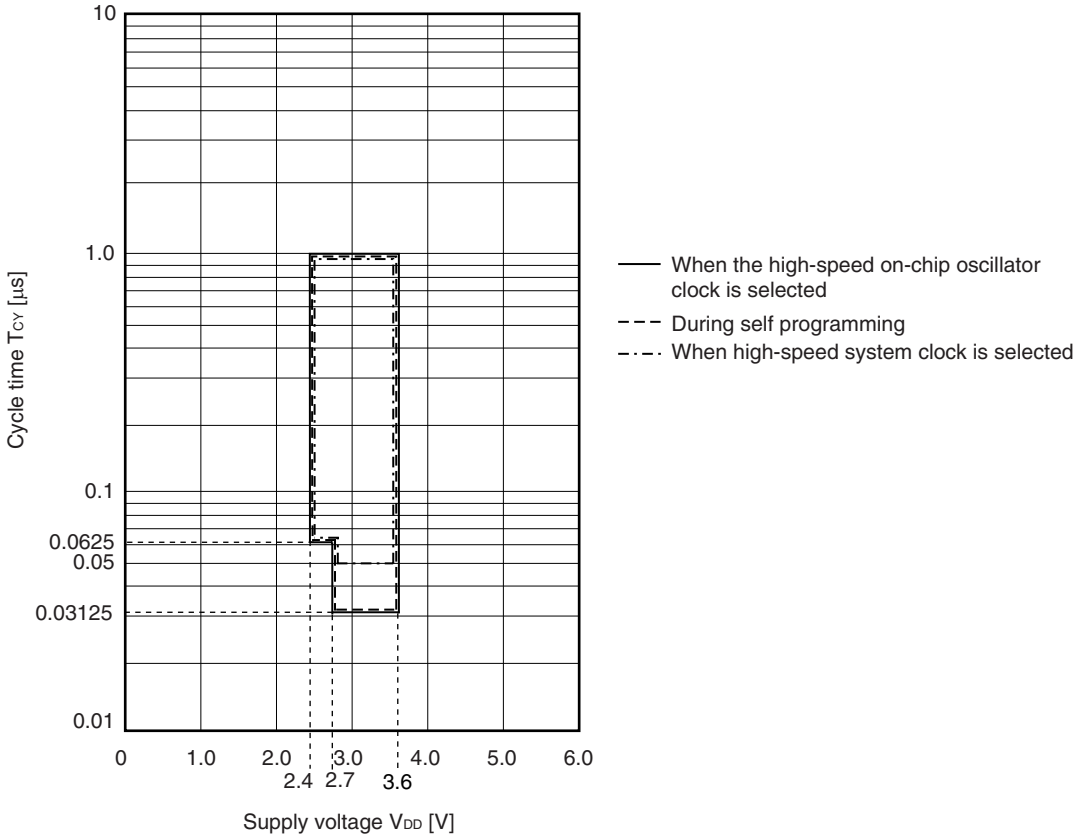
**Note** The following conditions are required for low-voltage interface when $EV_{DD0} < V_{DD}$.
2.4 V ≤ $EV_{DD0}$ < 2.7 V : MIN. 125 ns

**Remark** $f_{MCK}$: Timer array unit operation clock frequency
(Operation clock to be set by the CKS0n bit of timer clock select register 0 (TPS0) and timer mode register 0n (TMR0n). n: Channel number (n = 0 to 7))

**Minimum Instruction Execution Time during Main System Clock Operation**

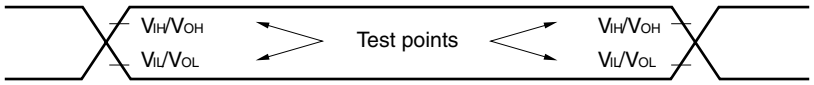
<R>

$T_{CY}$ vs $V_{DD}$ (HS (high-speed main) mode)

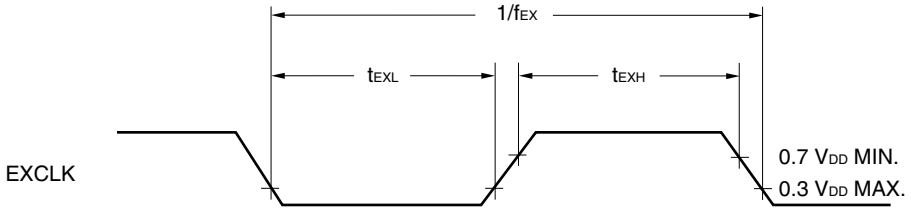


### AC Timing Test Points

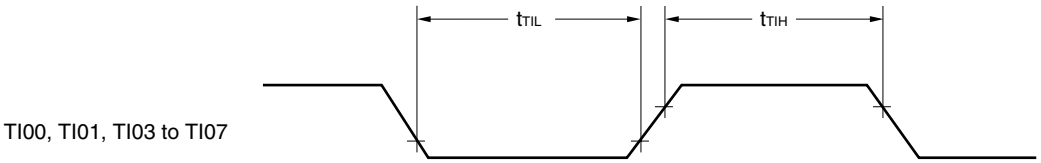
<R>

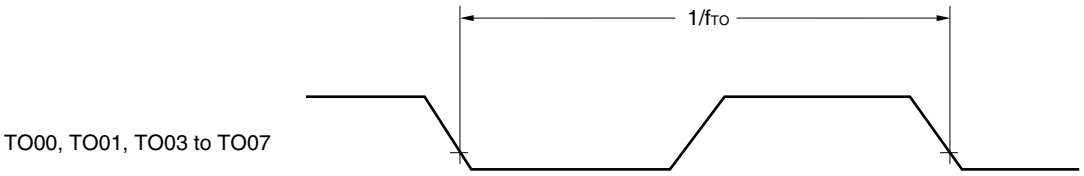



### External System Clock Timing

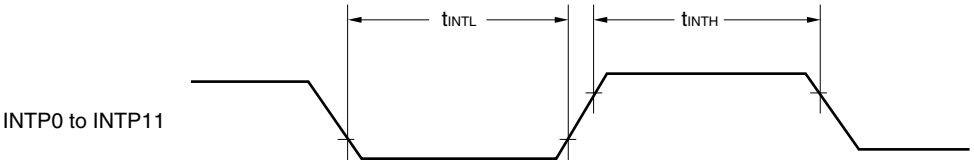



### <R> TI/TO Timing

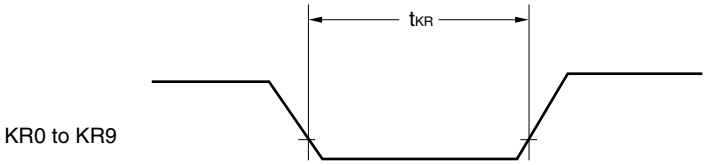


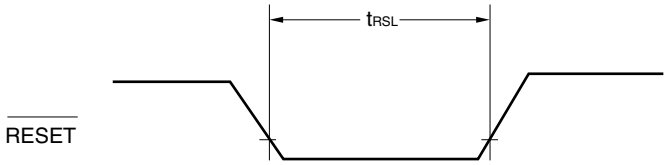

### Interrupt Request Input Timing




### Key Interrupt Input Timing



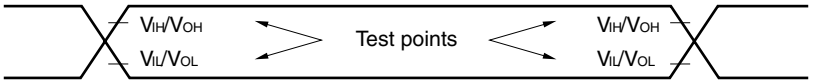
**$\overline{\text{RESET}}$ Input Timing**



## 30.5 Peripheral Functions Characteristics

### AC Timing Test Points

<R>




### 30.5.1 Serial array unit

**(1) During communication at same potential (UART mode) (dedicated baud rate generator output)**

**($T_A$ = −40 to +105°C, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

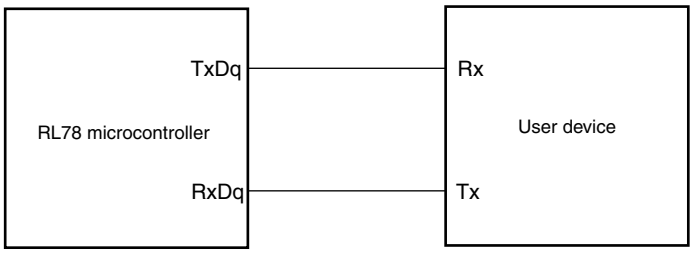
| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Transfer rate[Note 1] | | | | | $f_{MCK}$/12 | bps |
| | | Theoretical value of the maximum transfer rate $f_{CLK}$ = 32 MHz, $f_{MCK}$ = $f_{CLK}$ | | | 2.6[Note 2] | Mbps |

**Notes 1.** Transfer rate in the SNOOZE mode is 4800 bps.

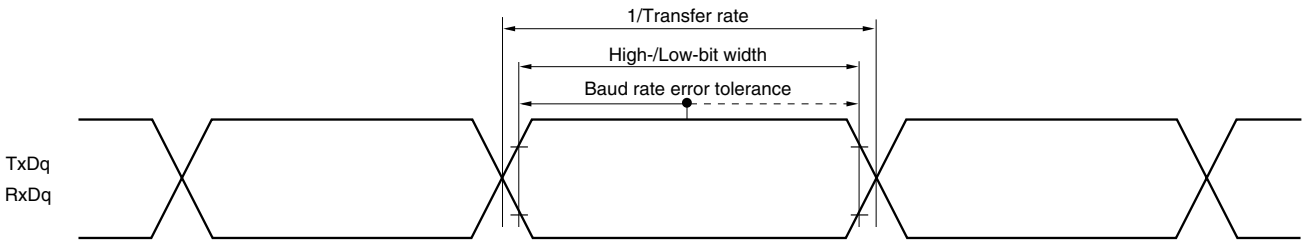
**2.** The following conditions are required for low-voltage interface when $EV_{DD0}$ < $V_{DD}$.
2.4 V ≤ $EV_{DD0}$ < 2.7 V : MAX. 1.3 Mbps

**Caution Select the normal input buffer for the RxDq pin and the normal output mode for the TxDq pin by using port input mode register g (PIMg) and port output mode register g (POMg).**

**UART mode connection diagram (during communication at same potential)**




**UART mode bit width (during communication at same potential) (reference)**




**Remarks 1.** q: UART number (q = 0 to 2), g: PIM and POM number (g = 0, 1)

**2.** $f_{MCK}$: Serial array unit operation clock frequency
(Operation clock to be set by the CKSmn bit of serial mode register mn (SMRmn). m: Unit number, n: Channel number (mn = 00 to 03, 10, 11))

**(2) During communication at same potential (CSI mode) (master mode, SCKp... internal clock output)**

**($T_A$ = −40 to +105°C, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| SCKp cycle time | $t_{KCY1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | $t_{KCY1} \geq 4/f_{CLK}$ | 250 | | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V | $t_{KCY1} \geq 4/f_{CLK}$ | 500 | | | ns |
| SCKp high-/low-level width | $t_{KH1}$, $t_{KL1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | $t_{KCY1}/2 - 36$ | | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V | | $t_{KCY1}/2 - 76$ | | | ns |
| SIp setup time (to SCKp↑)[Note 1] | $t_{SIK1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 66 | | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 113 | | | ns |
| SIp hold time (from SCKp↑)[Note 1] | $t_{KSI1}$ | | | 38 | | | ns |
| Delay time from SCKp↓ to SOp output[Note 2] | $t_{KSO1}$ | C = 30 p[Note 3] | | | | 50 | ns |

**Notes 1.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The SIp setup time or SIp hold time becomes "from SCKp↓" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

**2.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The delay time to SOp output becomes "from SCKp↑" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

**3.** C is the load capacitance of the SCKp and SOp output lines.

**Caution Select the normal input buffer for the SIp pin and the normal output mode for the SOp pin and SCKp pin by using port input mode register g (PIMg) and port output mode register g (POMg).**

**Remark** p: CSI number (p = 00, 01, 10, 11, 20, 21), m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), g: PIM and POM numbers (g = 0, 1)

<R> **(3) During communication at same potential (CSI mode) (slave mode, SCKp... external clock input)**

**($T_A$ = −40 to +105°C, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| SCKp cycle time[Note 1] | $t_{KCY2}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | 16 MHz < $f_{MCK}$ | 16/$f_{MCK}$ | | | ns |
| | | | $f_{MCK}$ ≤ 16 MHz | 12/$f_{MCK}$ | | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 12/$f_{MCK}$ and 1000 | | | ns |
| SCKp high-/low-level width | $t_{KH2}$, $t_{KL2}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | $t_{KCY2}$/2−14 | | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V | | $t_{KCY2}$/2−16 | | | ns |
| SIp setup time (to SCKp↑)[Note 2] | $t_{SIK2}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 1/$f_{MCK}$ + 40 | | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 1/$f_{MCK}$ + 60 | | | ns |
| SIp hold time (from SCKp↑)[Note 2] | $t_{KSI2}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 1/$f_{MCK}$+62 | | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V | | 1/$f_{MCK}$+62 | | | ns |
| Delay time from SCKp↓ to SOp output[Note 3] | $t_{KSO2}$ | C = 30 pF[Note 4] | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | 2/$f_{MCK}$+66 | ns |
| | | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | 2/$f_{MCK}$+113 | ns |

**Notes 1.** Transfer rate in the SNOOZE mode : MAX. 1 Mbps

**2.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The SIp setup time or SIp hold time becomes "from SCKp↓" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

**3.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The delay time to SOp output becomes "from SCKp↑" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

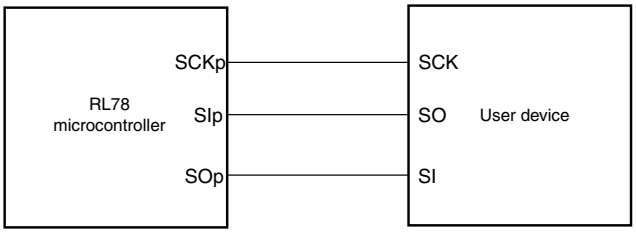
**4.** C is the load capacitance of the SOp output lines.

**Caution Select the normal input buffer for the SIp pin and SCKp pin and the normal output mode for the SOp pin by using port input mode register g (PIMg) and port output mode register g (POMg).**

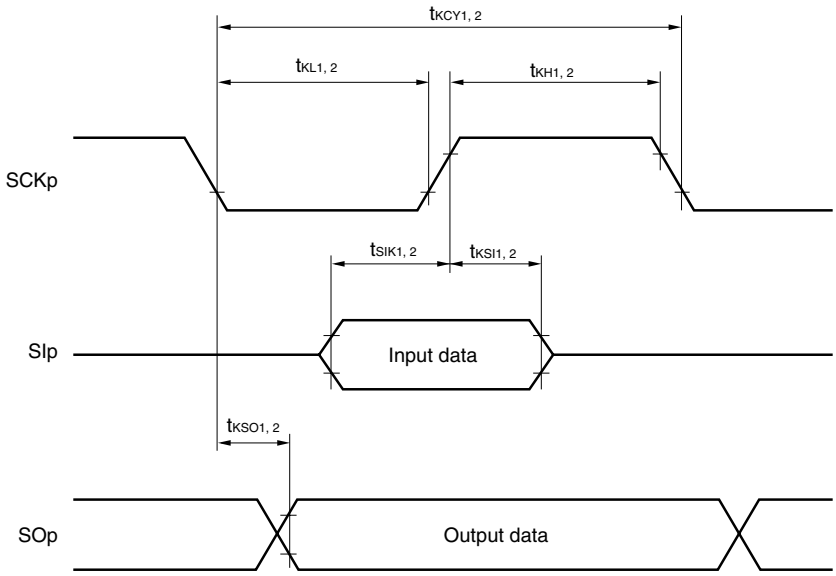
**Remarks 1.** p: CSI number (p = 00, 01, 10, 11, 20, 21), m: Unit number (m = 0, 1), n: Channel number (n = 0 to 3), g: PIM number (g = 0, 1)

**2.** $f_{MCK}$: Serial array unit operation clock frequency
(Operation clock to be set by the CKSmn bit of serial mode register mn (SMRmn). m: Unit number, n: Channel number (mn = 00 to 03, 10, 11))

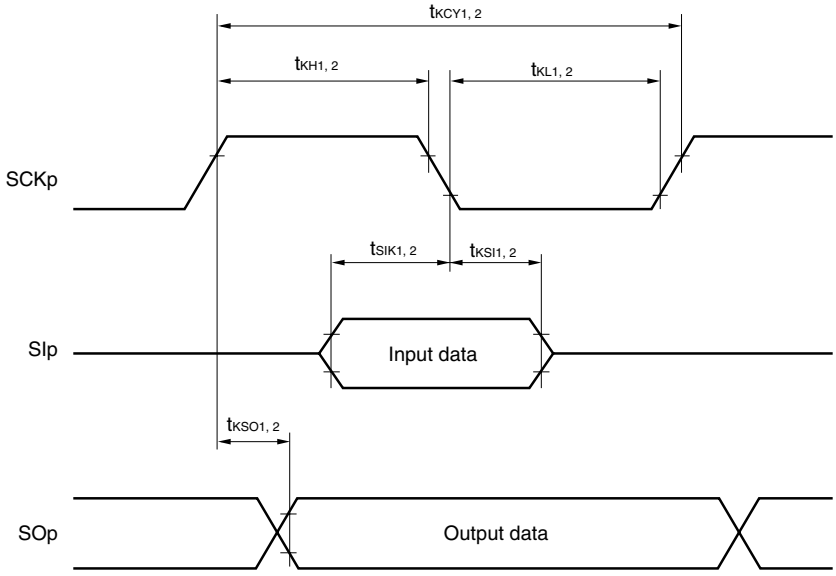
**CSI mode connection diagram (during communication at same potential)**




**CSI mode serial transfer timing (during communication at same potential)**
**(When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1.)**




**CSI mode serial transfer timing (during communication at same potential)**
**(When DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.)**




**Remarks 1.** p: CSI number (p = 00, 01, 10, 11, 20, 21)

**2.** m: Unit number, n: Channel number (mn = 00 to 03, 10, 11)

### (4) During communication at same potential (simplified $I^2C$ mode)

**($T_A$ = −40 to +105°C, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

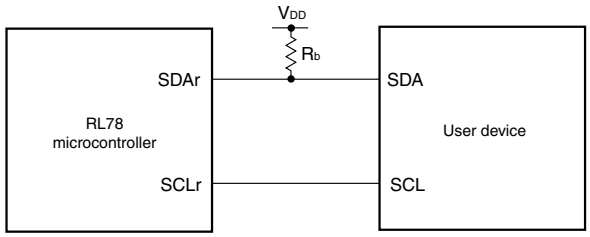
| Parameter | Symbol | Conditions | MIN. | MAX. | Unit |
|---|---|---|---|---|---|
| SCLr clock frequency | $f_{SCL}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | | 400[Note 1] | kHz |
| | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V, $C_b$ = 100 pF, $R_b$ = 3 kΩ | | 100[Note 1] | kHz |
| Hold time when SCLr = “L” | $t_{LOW}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 1200 | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V, $C_b$ = 100 pF, $R_b$ = 3 kΩ | 4600 | | ns |
| Hold time when SCLr = “H” | $t_{HIGH}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 1200 | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V, $C_b$ = 100 pF, $R_b$ = 3 kΩ | 4600 | | ns |
| Data setup time (reception) | $t_{SU:DAT}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 1/$f_{MCK}$ + 220[Note 2] | | ns |
| | | 2.4 V ≤ $EV_{DD}$ ≤ 3.6 V, $C_b$ = 100 pF, $R_b$ = 3 kΩ | 1/$f_{MCK}$ + 580[Note 2] | | ns |
| Data hold time (transmission) | $t_{HD:DAT}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 0 | 770 | ns |
| | | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V, $C_b$ = 100 pF, $R_b$ = 3 kΩ | 0 | 1420 | ns |

**Notes 1.** The value must also be $f_{CLK}$/4 or lower.

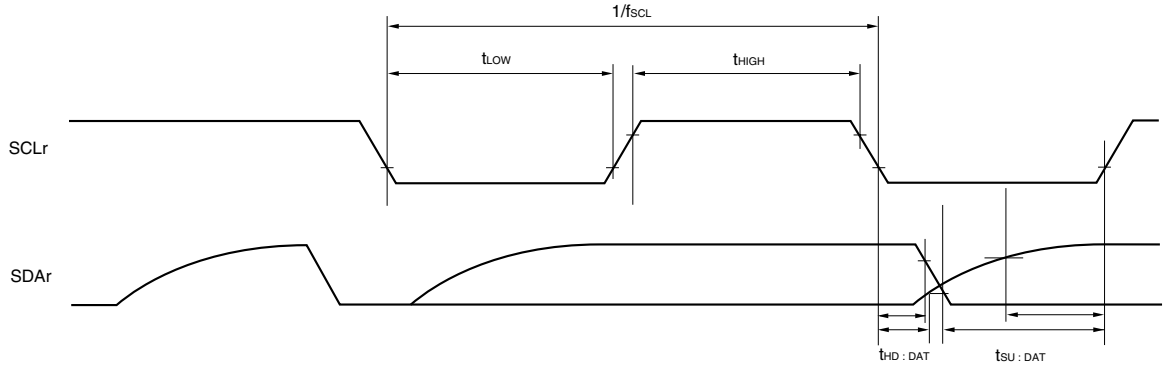
**2.** Set the $f_{MCK}$ value to keep the hold time of SCLr = “L” and SCLr = “H”.

**Caution Select the normal input buffer and the N-ch open drain output ($V_{DD}$ tolerance (When 25- to 48-pin products)/$EV_{DD}$ tolerance (When 64-pin products)) mode for the SDAr pin and the normal output mode for the SCLr pin by using port input mode register g (PIMg) and port output mode register h (POMh).**

**Simplified I²C mode mode connection diagram (during communication at same potential)**




**Simplified I²C mode serial transfer timing (during communication at same potential)**




**Remarks 1.** $R_b$[Ω]: Communication line (SDAr) pull-up resistance, $C_b$[F]: Communication line (SDAr, SCLr) load capacitance

**2.** r: IIC number (r = 00, 01, 10, 11, 20, 21), g: PIM number (g = 0, 1), h: POM number (h = 0, 1)

**3.** $f_{MCK}$: Serial array unit operation clock frequency
(Operation clock to be set by the CKSmn bit of serial mode register mn (SMRmn). m: Unit number, n: Channel number, mn = 00 to 03, 10, 11)

**(5) Communication at different potential (1.8 V, 2.5 V) (UART mode) (dedicated baud rate generator output) (1/2)**

**($T_A$ = −40 to +105°C, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|---|
| Transfer rate[Note 1] | | Reception | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V | | | | $f_{MCK}$/12 | bps |
| | | | | Theoretical value of the maximum transfer rate $f_{CLK}$ = 32 MHz, $f_{MCK}$ = $f_{CLK}$ | | | 2.6 | Mbps |
| | | | 2.4 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V | | | | $f_{MCK}$/12 | bps |
| | | | | Theoretical value of the maximum transfer rate $f_{CLK}$ = 32 MHz, $f_{MCK}$ = $f_{CLK}$ | | | 2.6[Note 2] | Mbps |

**Notes 1.** Transfer rate in the SNOOZE mode is 4800 bps.

**2.** The following conditions are required for low-voltage interface when $EV_{DD0}$ < $V_{DD}$.
2.4 V ≤ $EV_{DD0}$ < 2.7 V : MAX. 1.3 Mbps

**Caution Select the TTL input buffer for the RxDq pin and the N-ch open drain output ($V_{DD}$ tolerance (When 25- to 48-pin products)/$EV_{DD}$ tolerance (When 64-pin products)) mode for the TxDq pin by using port input** <R> **mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

**Remarks 1.** $V_b$[V]: Communication line voltage

**2.** q: UART number (q = 0 to 2), g: PIM and POM number (g = 0, 1)

**3.** $f_{MCK}$: Serial array unit operation clock frequency
(Operation clock to be set by the CKSmn bit of serial mode register mn (SMRmn). m: Unit number, n: Channel number (mn = 00 to 03, 10, 11)

**(5) Communication at different potential (1.8 V, 2.5 V) (UART mode) (dedicated baud rate generator output) (2/2)**
**($T_A$ = −40 to +105°C, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|---|
| Transfer rate | | Transmission | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V | | | | **Note 1** | bps |
| | | | | Theoretical value of the maximum transfer rate $C_b$ = 50 pF, $R_b$ = 2.7 kΩ, $V_b$ = 2.3 V | | | 1.2 Note 2 | Mbps |
| | | | 2.4 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V | | | | **Note 3** | bps |
| | | | | Theoretical value of the maximum transfer rate $C_b$ = 50 pF, $R_b$ = 5.5 kΩ, $V_b$ = 1.6 V | | | 0.43 Note 4 | Mbps |

**Notes 1.** The smaller maximum transfer rate derived by using $f_{MCK}$/12 or the following expression is the valid maximum transfer rate.
Expression for calculating the transfer rate when 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V and 2.3 V ≤ $V_b$ ≤ 2.7 V

$$\text{Maximum transfer rate} = \frac{1}{\{-C_b \times R_b \times \ln(1 - \frac{2.0}{V_b})\} \times 3} \text{ [bps]}$$

$$\text{Baud rate error (theoretical value)} = \frac{\frac{1}{\text{Transfer rate} \times 2} - \{-C_b \times R_b \times \ln(1 - \frac{2.0}{V_b})\}}{(\frac{1}{\text{Transfer rate}}) \times \text{Number of transferred bits}} \times 100 \text{ [\%]}$$

* This value is the theoretical value of the relative difference between the transmission and reception sides.

**2.** This value as an example is calculated when the conditions described in the “Conditions” column are met. See **Note 1** above to calculate the maximum transfer rate under conditions of the customer.

**3.** The smaller maximum transfer rate derived by using $f_{MCK}$/12 or the following expression is the valid maximum transfer rate.
Expression for calculating the transfer rate when 2.4 V ≤ $EV_{DD0}$ < 3.3 V and 1.6 V ≤ $V_b$ ≤ 2.0 V

$$\text{Maximum transfer rate} = \frac{1}{\{-C_b \times R_b \times \ln(1 - \frac{1.5}{V_b})\} \times 3} \text{ [bps]}$$

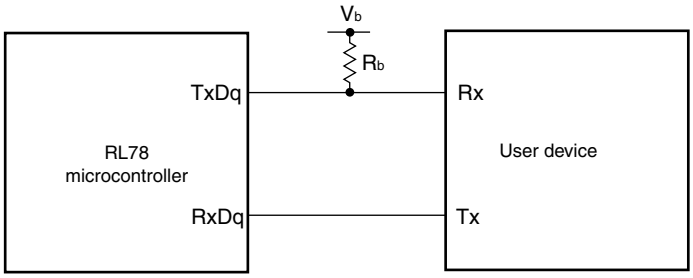
$$\text{Baud rate error (theoretical value)} = \frac{\frac{1}{\text{Transfer rate} \times 2} - \{-C_b \times R_b \times \ln(1 - \frac{1.5}{V_b})\}}{(\frac{1}{\text{Transfer rate}}) \times \text{Number of transferred bits}} \times 100 \text{ [\%]}$$

* This value is the theoretical value of the relative difference between the transmission and reception sides.

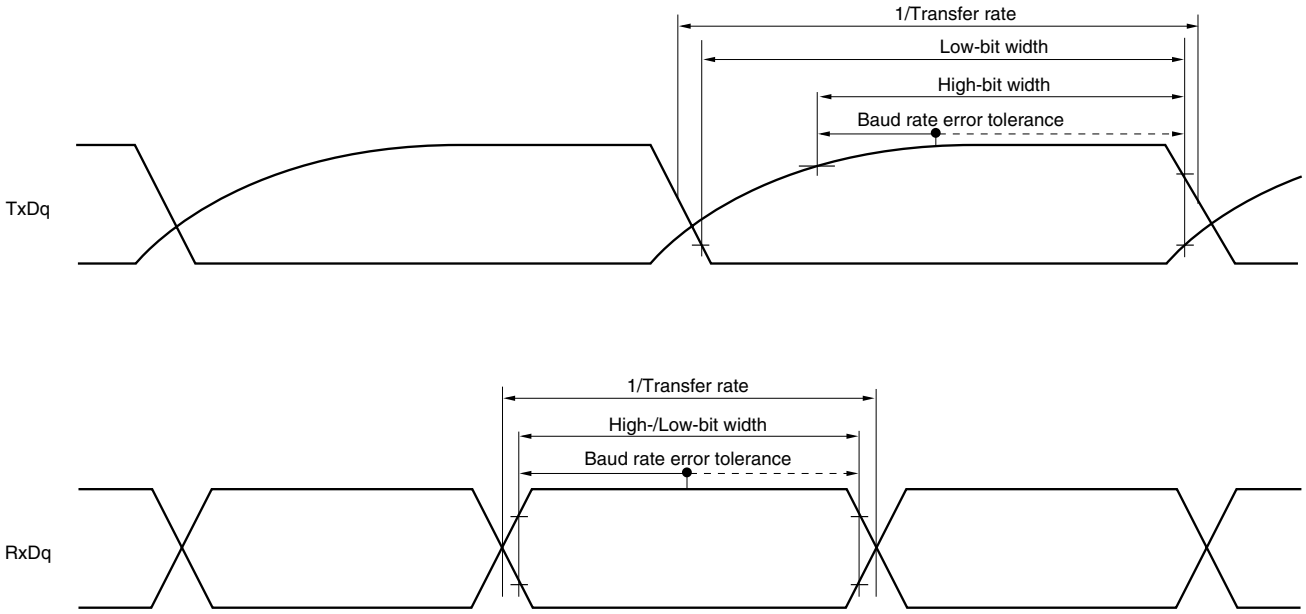
**4.** This value as an example is calculated when the conditions described in the “Conditions” column are met. See **Note 3** above to calculate the maximum transfer rate under conditions of the customer.

**Caution Select the TTL input buffer for the RxDq pin and the N-ch open drain output ($V_{DD}$ tolerance (When 25- to 48-pin products)/$EV_{DD}$ tolerance (When 64-pin products)) mode for the TxDq pin by using port input mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.** <R>

**UART mode connection diagram (during communication at different potential)**




**UART mode bit width (during communication at different potential) (reference)**




**Remarks 1.** $R_b$[Ω]: Communication line (TxDq) pull-up resistance,
$C_b$[F]: Communication line (TxDq) load capacitance, $V_b$[V]: Communication line voltage

**2.** q: UART number (q = 0 to 2), g: PIM and POM number (g = 0, 1)

**3.** $f_{MCK}$: Serial array unit operation clock frequency
(Operation clock to be set by the CKSmn bit of serial mode register mn (SMRmn).
m: Unit number, n: Channel number (mn = 00 to 03, 10, 11))

**(6) Communication at different potential (1.8 V, 2.5 V) (CSI mode) (master mode, SCKp... internal clock output) (1/2)**

**($T_A$ = −40 to +105°C, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| SCKp cycle time | $t_{KCY1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V,<br>2.3 V ≤ $V_b$ ≤ 2.7 V,<br>$C_b$ = 30 pF, $R_b$ = 2.7 kΩ | $t_{KCY1}$ ≥ 4/$f_{CLK}$ | 1000 | | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ < 3.3 V,<br>1.6 V ≤ $V_b$ ≤ 2.0 V,<br>$C_b$ = 30 pF, $R_b$ = 5.5 kΩ | $t_{KCY1}$ ≥ 4/$f_{CLK}$ | 2300 | | | ns |
| SCKp high-level width | $t_{KH1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V,<br>$C_b$ = 30 pF, $R_b$ = 2.7 kΩ | | $t_{KCY1}$/2 − 340 | | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V,<br>$C_b$ = 30 pF, $R_b$ = 5.5 kΩ | | $t_{KCY1}$/2 − 916 | | | ns |
| SCKp low-level width | $t_{KL1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V,<br>$C_b$ = 30 pF, $R_b$ = 2.7 kΩ | | $t_{KCY1}$/2 − 36 | | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V,<br>$C_b$ = 30 pF, $R_b$ = 5.5 kΩ | | $t_{KCY1}$/2 − 100 | | | ns |

**Caution Select the TTL input buffer for the SIp pin and the N-ch open drain output ($V_{DD}$ tolerance (When 25- to 48-pin products)/$EV_{DD}$ tolerance (When 64-pin products)) mode for the SOp pin and SCKp pin by using** <R> **port input mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

**Remarks 1.** $R_b$[Ω]: Communication line (SCKp, SOp) pull-up resistance, $C_b$[F]: Communication line (SCKp, SOp) load capacitance, $V_b$[V]: Communication line voltage

**2.** p: CSI number (p = 00, 10, 20), m: Unit number , n: Channel number (mn = 00, 02, 10), g: PIM and POM number (g = 0, 1)

**3.** CSI01, CSI11, and CSI21 cannot communicate at different potential. Use other CSI for communication at different potential.

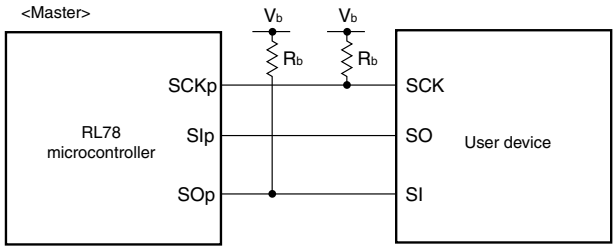
**(6) Communication at different potential (1.8 V, 2.5 V) (CSI mode) (master mode, SCKp... internal clock output) (2/2)**
**($T_A$ = −40 to +105°C, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| SIp setup time (to SCKp↑)[Note 1] | $t_{SIK1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | 354 | | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V, $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | 958 | | | ns |
| SIp hold time (from SCKp↑)[Note 1] | $t_{KSI1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | 38 | | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V, $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | 38 | | | ns |
| Delay time from SCKp↓ to SOp output[Note 1] | $t_{KSO1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | | | 390 | ns |
| | | 2.4 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V, $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | | | 966 | ns |
| SIp setup time (to SCKp↓)[Note 2] | $t_{SIK1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | 88 | | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V, $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | 220 | | | ns |
| SIp hold time (from SCKp↓)[Note 2] | $t_{KSI1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | 38 | | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V, $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | 38 | | | ns |
| Delay time from SCKp↑ to SOp output[Note 2] | $t_{KSO1}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | | | 50 | ns |
| | | 2.4 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V, $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | | | 50 | ns |

**Notes 1.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1.
**2.** When DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

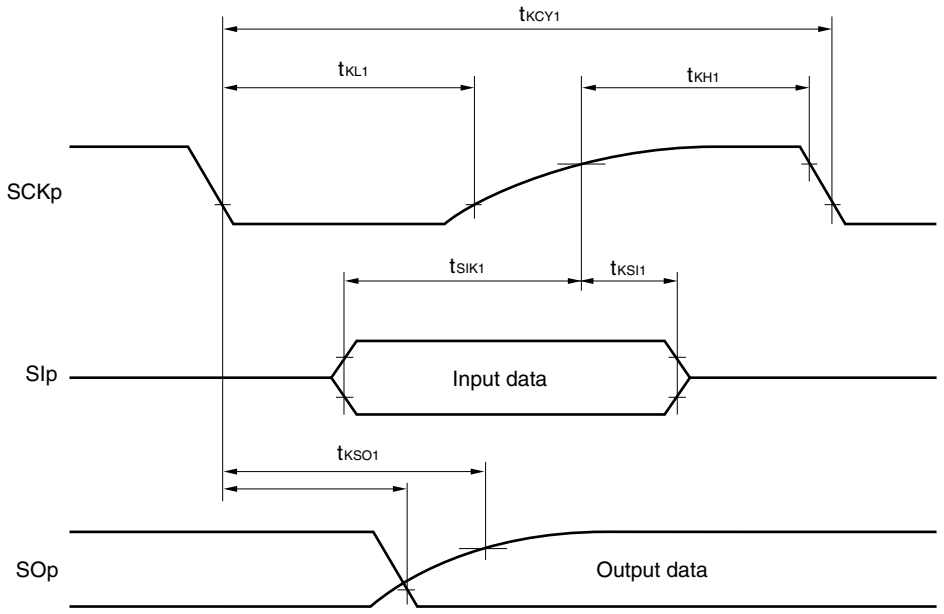
**Caution Select the TTL input buffer for the SIp pin and the N-ch open drain output ($V_{DD}$ tolerance (When 25- to 48-pin products)/$EV_{DD}$ tolerance (When 64-pin products)) mode for the SOp pin and SCKp pin by using** <R> **port input mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

**CSI mode connection diagram (during communication at different potential)**

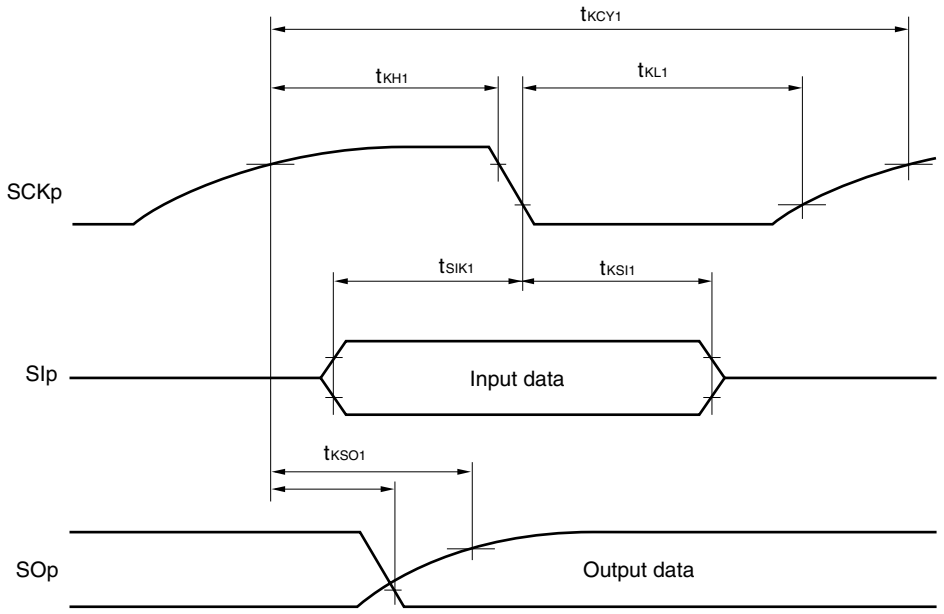



**Remarks 1.** $R_b$[Ω]: Communication line (SCKp, SOp) pull-up resistance, $C_b$[F]: Communication line (SCKp, SOp) load capacitance, $V_b$[V]: Communication line voltage
**2.** p: CSI number (p = 00, 10, 20), m: Unit number , n: Channel number (mn = 00, 02, 10), g: PIM and POM number (g = 0, 1)
**3.** CSI01, CSI11, and CSI21 cannot communicate at different potential. Use other CSI for communication at different potential.

**CSI mode serial transfer timing (master mode) (during communication at different potential)**
**(When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1.)**




**CSI mode serial transfer timing (master mode) (during communication at different potential)**
**(When DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.)**




**Remarks 1.** p: CSI number (p = 00, 10, 20), m: Unit number, n: Channel number (m = 00, 02, 10), g: PIM and POM number (g = 0, 1)

**2.** CSI01, CSI11, and CSI21 cannot communicate at different potential. Use other CSI for communication at different potential.

**(7) Communication at different potential (1.8 V, 2.5 V) (CSI mode) (slave mode, SCKp... external clock input)**

**($T_A$ = −40 to +105°C, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| SCKp cycle time[Note 1] | $t_{KCY2}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V | 24 MHz < $f_{MCK}$ | 40/$f_{MCK}$ | | | ns |
| | | | 20 MHz < $f_{MCK}$ ≤ 24 MHz | 32/$f_{MCK}$ | | | ns |
| | | | 16 MHz < $f_{MCK}$ ≤ 20 MHz | 28/$f_{MCK}$ | | | ns |
| | | | 8 MHz < $f_{MCK}$ ≤ 16 MHz | 24/$f_{MCK}$ | | | ns |
| | | | 4 MHz < $f_{MCK}$ ≤ 8 MHz | 16/$f_{MCK}$ | | | ns |
| | | | $f_{MCK}$ ≤ 4 MHz | 12/$f_{MCK}$ | | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V | 24 MHz < $f_{MCK}$ | 96/$f_{MCK}$ | | | ns |
| | | | 20 MHz < $f_{MCK}$ ≤ 24 MHz | 72/$f_{MCK}$ | | | ns |
| | | | 16 MHz < $f_{MCK}$ ≤ 20 MHz | 64/$f_{MCK}$ | | | ns |
| | | | 8 MHz < $f_{MCK}$ ≤ 16 MHz | 52/$f_{MCK}$ | | | ns |
| | | | 4 MHz < $f_{MCK}$ ≤ 8 MHz | 32/$f_{MCK}$ | | | ns |
| | | | $f_{MCK}$ ≤ 4 MHz | 20/$f_{MCK}$ | | | ns |
| SCKp high-/low-level width | $t_{KH2}$, $t_{KL2}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V | | $t_{KCY2}$/2 − 36 | | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V | | $t_{KCY2}$/2 − 100 | | | ns |
| SIp setup time (to SCKp↑)[Note 2] | $t_{SIK2}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V | | 1/$f_{MCK}$ + 40 | | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V | | 1/$f_{MCK}$ + 60 | | | |
| SIp hold time (from SCKp↑)[Note 2] | $t_{KSI2}$ | | | 1/$f_{MCK}$ + 62 | | | ns |
| Delay time from SCKp↓ to SOp output[Note 3] | $t_{KSO2}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 30 pF, $R_b$ = 2.7 kΩ | | | | 2/$f_{MCK}$ + 428 | ns |
| | | 2.4 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V, $C_b$ = 30 pF, $R_b$ = 5.5 kΩ | | | | 2/$f_{MCK}$ + 1146 | ns |

**Notes 1.** Transfer rate in the SNOOZE mode : MAX. 1 Mbps

**2.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The SIp setup time or SIp hold time becomes "from SCKp↓" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

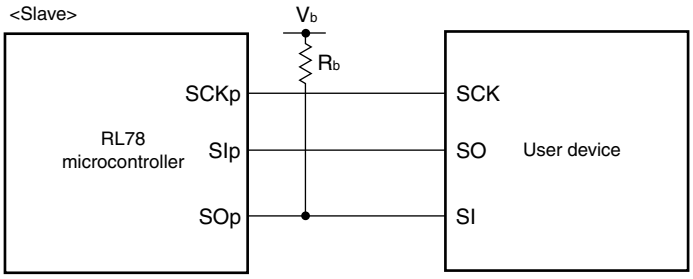
**3.** When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The delay time to SOp output becomes "from SCKp↑" when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.

**Caution Select the TTL input buffer for the SIp pin and SCKp pin and the N-ch open drain output ($V_{DD}$ tolerance (When 25- to 48-pin products)/$EV_{DD}$ tolerance (When 64-pin products)) mode for the SOp pin by using port input mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

<R>

(**Remarks** are listed on the next page.)

**CSI mode connection diagram (during communication at different potential)**



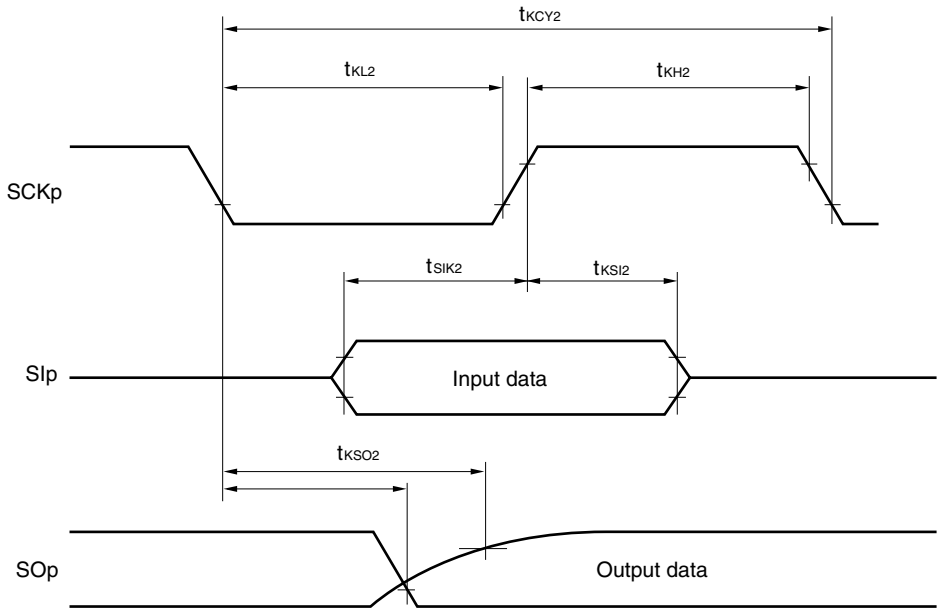

**Remarks 1.** $R_b$[Ω]: Communication line (SOp) pull-up resistance, $C_b$[F]: Communication line (SOp) load capacitance, $V_b$[V]: Communication line voltage

**2.** p: CSI number (p = 00, 10, 20), m: Unit number (m = 0, 1), n: Channel number (n = 00, 02, 10), g: PIM and POM number (g = 0, 1)

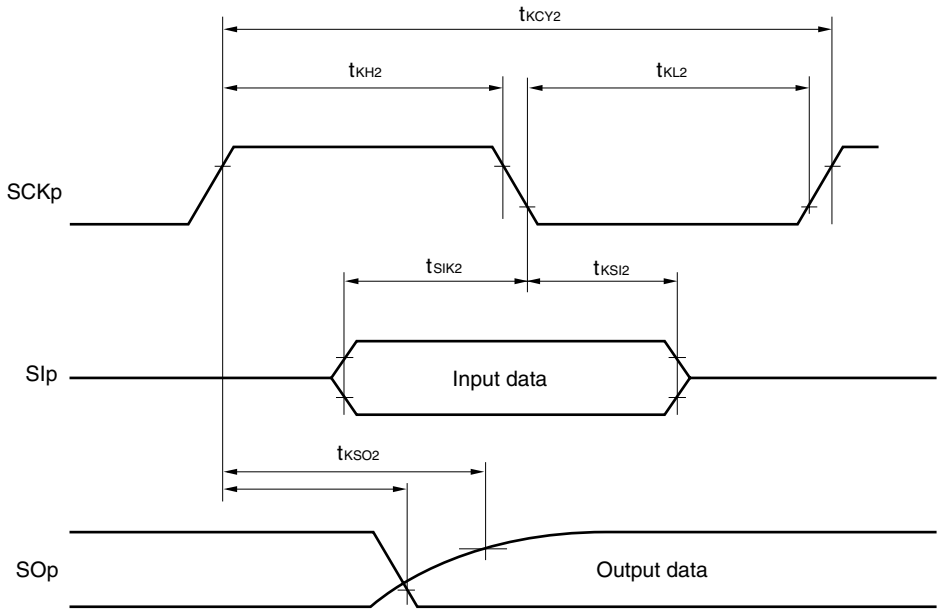
**3.** $f_{MCK}$: Serial array unit operation clock frequency
(Operation clock to be set by the CKSmn bit of serial mode register mn (SMRmn).
m: Unit number, n: Channel number (mn = 00, 02, 10))

**4.** CSI01, CSI11, and CSI21 cannot communicate at different potential. Use other CSI for communication at different potential.

**CSI mode serial transfer timing (slave mode) (during communication at different potential)**
**(When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1.)**




**CSI mode serial transfer timing (slave mode) (during communication at different potential)**
**(When DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.)**




**Remarks 1.** p: CSI number (p = 00, 10, 20), m: Unit number, n: Channel number (mn = 00, 02, 10), g: PIM and POM number (g = 0, 1)

**2.** CSI01, CSI11, and CSI21 cannot communicate at different potential. Use other CSI for communication at different potential.

**(8) Communication at different potential (1.8 V, 2.5 V) (simplified $I^2C$ mode) (1/2)**

**($T_A$ = −40 to +105°C, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | MAX. | Unit |
|---|---|---|---|---|---|
| SCLr clock frequency | $f_{SCL}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | | 400[Note 1] | kHz |
| | | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 100 pF, $R_b$ = 2.7 kΩ | | 100[Note 1] | kHz |
| | | 2.4 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V, $C_b$ = 100 pF, $R_b$ = 5.5 kΩ | | 100[Note 1] | kHz |
| Hold time when SCLr = “L” | $t_{LOW}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 1200 | | ns |
| | | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 100 pF, $R_b$ = 2.7 kΩ | 4600 | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V, $C_b$ = 100 pF, $R_b$ = 5.5 kΩ | 4650 | | ns |
| Hold time when SCLr = “H” | $t_{HIGH}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 500 | | ns |
| | | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 100 pF, $R_b$ = 2.7 kΩ | 2400 | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V, $C_b$ = 100 pF, $R_b$ = 5.5 kΩ | 1830 | | ns |

(**Notes**, **Caution** and **Remarks** are listed on the next page.)

**(8) Communication at different potential (1.8 V, 2.5 V) (simplified $I^2C$ mode) (2/2)**

**($T_A$ = −40 to +105°C, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | MAX. | Unit |
|---|---|---|---|---|---|
| Data setup time (reception) | $t_{SU:DAT}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | $1/f_{MCK}$ + 340 Note 2 | | ns |
| | | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 100 pF, $R_b$ = 2.7 kΩ | $1/f_{MCK}$ + 760 Note 2 | | ns |
| | | 2.4 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V, $C_b$ = 100 pF, $R_b$ = 5.5 kΩ | $1/f_{MCK}$ + 570 Note 2 | | ns |
| Data hold time (transmission) | $t_{HD:DAT}$ | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 50 pF, $R_b$ = 2.7 kΩ | 0 | 770 | ns |
| | | 2.7 V ≤ $EV_{DD0}$ ≤ 3.6 V, 2.3 V ≤ $V_b$ ≤ 2.7 V, $C_b$ = 100 pF, $R_b$ = 2.7 kΩ | 0 | 1420 | ns |
| | | 2.4 V ≤ $EV_{DD0}$ < 3.3 V, 1.6 V ≤ $V_b$ ≤ 2.0 V, $C_b$ = 100 pF, $R_b$ = 5.5 kΩ | 0 | 1215 | ns |

**Notes 1.** The value must also be $f_{CLK}$/4 or lower.

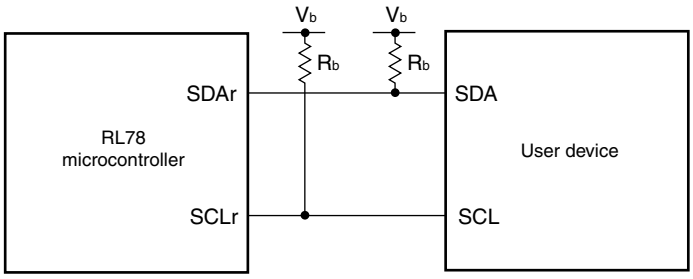
**2.** Set the $f_{MCK}$ value to keep the hold time of SCLr = "L" and SCLr = "H".

**Caution Select the TTL input buffer and the N-ch open drain output ($V_{DD}$ tolerance (When 25- to 48-pin products)/$EV_{DD}$ tolerance (When 64-pin products)) mode for the SDAr pin and the N-ch open drain output ($V_{DD}$ tolerance (When 25- to 48-pin products)/$EV_{DD}$ tolerance (When 64-pin products)) mode for the SCLr pin by using port input mode register g (PIMg) and port output mode register g (POMg). For $V_{IH}$ and $V_{IL}$, see the DC characteristics with TTL input buffer selected.**

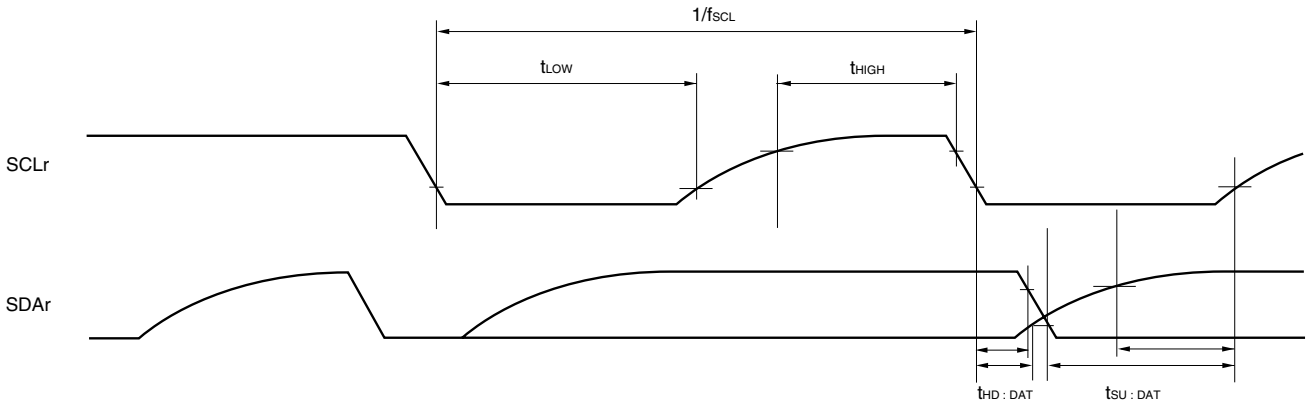
<R>

(**Remarks** are listed on the next page.)

**Simplified I²C mode connection diagram (during communication at different potential)**




**Simplified I²C mode serial transfer timing (during communication at different potential)**




**Remarks 1.** $R_b$[Ω]: Communication line (SDAr, SCLr) pull-up resistance, $C_b$[F]: Communication line (SDAr, SCLr) load capacitance, $V_b$[V]: Communication line voltage

**2.** r: IIC number (r = 00, 10, 20), g: PIM, POM number (g = 0, 1)

**3.** $f_{MCK}$: Serial array unit operation clock frequency
(Operation clock to be set by the CKSmn bit of serial mode register mn (SMRmn). m: Unit number, n: Channel number (mn = 00, 02, 10)

**4.** IIC01, IIC11, and IIC21 cannot communicate at different potential. Use IIC00, IIC10, or IIC20 for communication at different potential.

### 30.5.2 Serial interface IICA

#### (1) $I^2C$ standard mode, fast mode

**($T_A$ = −40 to +105°C, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| Parameter | Symbol | Conditions | | Standard Mode | | Fast Mode | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | | | MIN. | MAX. | MIN. | MAX. | |
| SCLA0 clock frequency | $f_{SCL}$ | Fast mode: $f_{CLK}$ ≥ 3.5 MHz | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V | | | 0 | 400 | kHz |
| | | Normal mode: $f_{CLK}$ ≥ 1 MHz | 2.4 V ≤ $EV_{DD0}$ ≤ 3.6 V | 0 | 100 | | | kHz |
| Setup time of restart condition | $t_{SU:STA}$ | | | 4.7 | | 0.6 | | μs |
| Hold time[Note 1] | $t_{HD:STA}$ | | | 4.0 | | 0.6 | | μs |
| Hold time when SCLA0 = "L" | $t_{LOW}$ | | | 4.7 | | 1.3 | | μs |
| Hold time when SCLA0 = "H" | $t_{HIGH}$ | | | 4.0 | | 0.6 | | μs |
| Data setup time (reception) | $t_{SU:DAT}$ | | | 250 | | 100 | | ns |
| Data hold time (transmission)[Note 2] | $t_{HD:DAT}$ | | | 0 | 3.45 | 0 | 0.9 | μs |
| Setup time of stop condition | $t_{SU:STO}$ | | | 4.0 | | 0.6 | | μs |
| Bus-free time | $t_{BUF}$ | | | 4.7 | | 1.3 | | μs |

**Notes 1.** The first clock pulse is generated after this period when the start/restart condition is detected.

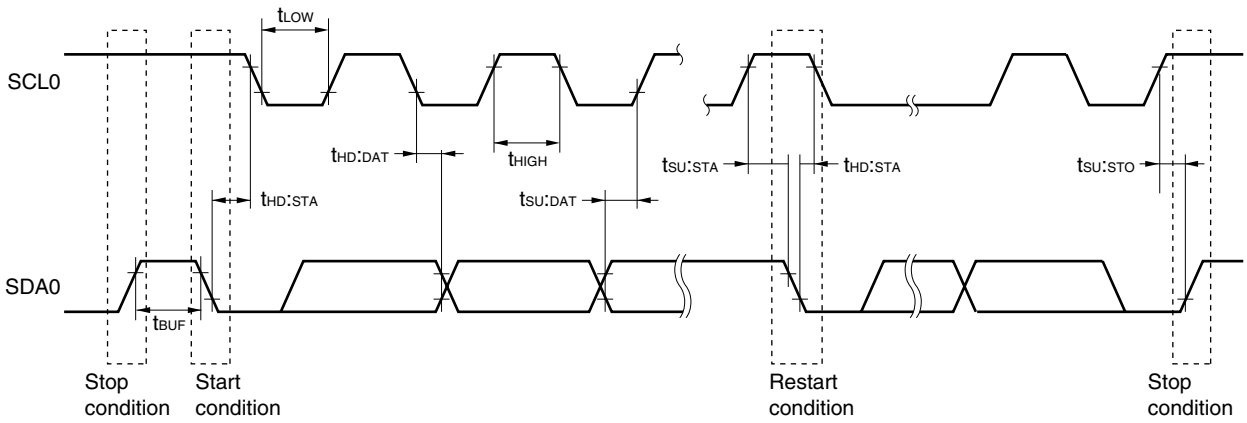
**2.** The maximum value (MAX.) of $t_{HD:DAT}$ is during normal transfer and a wait state is inserted in the $\overline{ACK}$ (acknowledge) timing.

**Remark** The maximum value of $C_b$ (communication line capacitance) and the value of $R_b$ (communication line pull-up resistor) at that time in each mode are as follows.

Standard mode: $C_b$ = 400 pF, $R_b$ = 2.7 kΩ

Fast mode: $C_b$ = 320 pF, $R_b$ = 1.1 kΩ

**IICA serial transfer timing**



## 30.6 Analog Characteristics

### 30.6.1 A/D converter characteristics

**Division of A/D Converter Characteristics**

| Reference voltag / Input channel | Reference voltage (+) = $AV_{REFP}$<br>Reference voltage (−) = $AV_{REFM}$ | Reference voltage (+) = $AV_{DD}$<br>Reference voltage (−) = $AV_{SS}$ | Reference voltage (+) = Internal refrence voltage<br>Reference voltage (−) = $AV_{SS}$ |
|---|---|---|---|
| High-accuracy channel; ANI0 to ANI12<br>(input buffer power supply: $AV_{DD}$) | See **30.6.1 (1)** | See **30.6.1 (2)** | See **30.6.1 (5)** |
| Standard channel; ANI16 to ANI30<br>(input buffer power supply: $V_{DD}$ or $EV_{DD0}$) | See **30.6.1 (3)** | See **30.6.1 (4)** | |
| Temperature sensor, internal reference voltage output | See **30.6.1 (3)** | See **30.6.1 (4)** | − |

<R> **(1) When reference voltage (+) = $AV_{REFP}$/ANI0 (ADREFP1 = 0, ADREFP0 = 1), reference voltage (−) = $AV_{REFM}$/ANI1 (ADREFM = 1), target for conversion: ANI2 to ANI12**

**($T_A$ = −40 to +105°C, 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = 0 V, $AV_{SS}$ = 0 V, Reference voltage (+) = $AV_{REFP}$, Reference voltage (−) = $AV_{REFM}$ = 0 V)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Resolution | $R_{ES}$ | | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 8. | | 12. | bit |
| Overall error[Note] | AINL | 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±6.0 | LSB |
| Conversion time | $t_{CONV}$ | ADTYP = 0,<br>12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 3.375 | | | μs |
| Zero-scale error[Note] | $E_{ZS}$ | 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±4.5 | LSB |
| Full-scale error[Note] | $E_{FS}$ | 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±4.5 | LSB |
| Integral linearity error[Note] | ILE | 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.0 | LSB |
| Differential linearity error[Note] | DLE | 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±1.5 | LSB |
| Analog input voltage | $V_{AIN}$ | | | 0 | | $AV_{REFP}$ | V |

**Note** Excludes quantization error (±1/2 LSB).

<R> **(2) When reference voltage (+) = $AV_{DD}$ (ADREFP1 = 0, ADREFP0 = 0), reference voltage (–) = $AV_{SS}$ (ADREFM = 0), target for conversion: ANI0 to ANI12**

**($T_A$ = −40 to +105°C, 2.4 V ≤ $AV_{DD}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = 0 V, $AV_{SS}$ = 0 V, Reference voltage (+) = $AV_{DD}$, Reference voltage (–) = $AV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Resolution | $R_{ES}$ | | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | 8 | | 12 | bit |
| Overall error[Note] | AINL | 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±7.5 | LSB |
| Conversion time | $t_{CONV}$ | ADTYP = 0,<br>12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | 3.375 | | | μs |
| Zero-scale error[Note] | $E_{ZS}$ | 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±6.0 | LSB |
| Full-scale error[Note] | $E_{FS}$ | 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±6.0 | LSB |
| Integral linearity error[Note] | ILE | 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±3.0 | LSB |
| Differential linearity error[Note] | DLE | 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.0 | LSB |
| Analog input voltage | $V_{AIN}$ | | | 0 | | $AV_{DD}$ | V |

**Note** Excludes quantization error (±1/2 LSB).

<R> **(3) When reference voltage (+) = $AV_{REFP}$/ANI0 (ADREFP1 = 0, ADREFP0 = 1), reference voltage (–) = $AV_{REFM}$/ANI1 (ADREFM = 1), target for conversion: ANI16 to ANI30, interanal reference voltage, temperature sensor output voltage**

**($T_A$ = −40 to +105°C, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V, $AV_{SS}$ = 0 V, Reference voltage (+) = $AV_{REFP}$, Reference voltage (–) = $AV_{REFM}$ = 0 V)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Resolution | $R_{ES}$ | | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 8 | | 12 | bit |
| Overall error[Note 1] | AINL | 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±7.0 | LSB |
| Conversion time | $t_{CONV}$ | ADTYP = 0, 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | 4.125 | | | μs |
| Zero-scale error[Note 1] | $E_{ZS}$ | 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±5.0 | LSB |
| Full-scale error[Note 1] | $E_{FS}$ | 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±5.0 | LSB |
| Integral linearity error[Note 1] | ILE | 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±3.0 | LSB |
| Differential linearity error[Note 1] | DLE | 12-bit resolution | 2.4 V ≤ $AV_{REFP}$ ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.0 | LSB |
| Analog input voltage | $V_{AIN}$ | | | 0. | | $AV_{REFP}$ and $EV_{DD0}$ | V |
| | | Interanal reference voltage (2.4 V ≤ $V_{DD}$ ≤ 3.6 V, HS (high-speed main) mode) | | $V_{BGR}$[Note 2] | | | V |
| | | Temperature sensor output voltage (2.4 V ≤ $V_{DD}$ ≤ 3.6 V, HS (high-speed main) mode) | | $V_{TMPS25}$[Note 2] | | | V |

**Notes 1.** Excludes quantization error (±1/2 LSB).

**2.** See **30.6.2 Temperature sensor, internal reference voltage output characteristics**.

<R> **(4) When reference voltage (+) = $AV_{DD}$ (ADREFP1 = 0, ADREFP0 = 0), reference voltage (–) = $AV_{SS}$ (ADREFM = 0), target for conversion: ANI16 to ANI30, interanal reference voltage, temperature sensor output voltage**

**($T_A$ = −40 to +105°C, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD0}$ ≤ 3.6 V, 2.4 V ≤ $AV_{DD}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V, $AV_{SS}$ = 0 V, Reference voltage (+) = $AV_{DD}$, Reference voltage (–) = $AV_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Resolution | $R_{ES}$ | | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | 8 | | 12 | bit |
| Overall error[Note 1] | AINL | 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±8.5 | LSB |
| Conversion time | $t_{CONV}$ | ADTYP = 0,<br>12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | 4.125 | | | μs |
| Zero-scale error[Note 1] | $E_{ZS}$ | 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±8.0 | LSB |
| Full-scale error[Note 1] | $E_{FS}$ | 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±8.0 | LSB |
| Integral linearity error[Note 1] | ILE | 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±3.5 | LSB |
| Differential linearity error[Note 1] | DLE | 12-bit resolution | 2.4 V ≤ $AV_{DD}$ ≤ 3.6 V | | | ±2.5 | LSB |
| Analog input voltage | $V_{AIN}$ | | | 0 | | $AV_{DD}$ and $EV_{DD0}$ | V |
| | | Interanal reference voltage<br>(2.4 V ≤ $V_{DD}$ ≤ 3.6 V, HS (high-speed main) mode) | | $V_{BGR}$[Note 2] | | | V |
| | | Temperature sensor output voltage<br>(2.4 V ≤ $V_{DD}$ ≤ 3.6 V, HS (high-speed main) mode) | | $V_{TMPS25}$[Note 2] | | | V |

**Notes 1.** Excludes quantization error (±1/2 LSB).

**2.** See **30.6.2 Temperature sensor, internal reference voltage output characteristics**.

<R> **(5) When reference voltage (+) = Internal reference voltage (1.45 V) (ADREFP1 = 1, ADREFP0 = 0), reference voltage (–) = $AV_{SS}$ (ADREFM = 0), target for conversion: ANI0 to ANI12, ANI16 to ANI30**

**($T_A$ = –40 to +105°C, 2.4 V ≤ $V_{DD}$ ≤ 3.6 V, 2.4 V ≤ $EV_{DD}$ ≤ $V_{DD}$, 2.4 V ≤ $AV_{DD}$ ≤ $V_{DD}$, $V_{SS}$ = $EV_{SS0}$ = 0 V, $AV_{SS}$ = 0 V, Reference voltage (+) = Internal reference voltage, Reference voltage (–) = $AV_{SS}$ = 0 V, HS (high-speed main) mode)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Resolution | $R_{ES}$ | | | 8 | | bit |
| Conversion time | $t_{CONV}$ | 8-bit resolution | 16.0 | | | μs |
| Zero-scale error[Note] | $E_{ZS}$ | 8-bit resolution | | | ±4.0 | LSB |
| Integral linearity error[Note] | ILE | 8-bit resolution | | | ±2.0 | LSB |
| Differential linearity error[Note] | DLE | 8-bit resolution | | | ±2.5 | LSB |
| Reference voltage (+) | $AV_{REF(+)}$ | = Internal reference voltage ($V_{BGR}$) | 1.38 | 1.45 | 1.50 | V |
| Analog input voltage | $V_{AIN}$ | | 0 | | $V_{BGR}$ | V |

**Note** Excludes quantization error (±1/2 LSB).

### 30.6.2 Temperature sensor, internal reference voltage output characteristics

**($T_A$ = –40 to +105°C, 2.4 V ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = 0 V, HS (high-speed main) mode)**

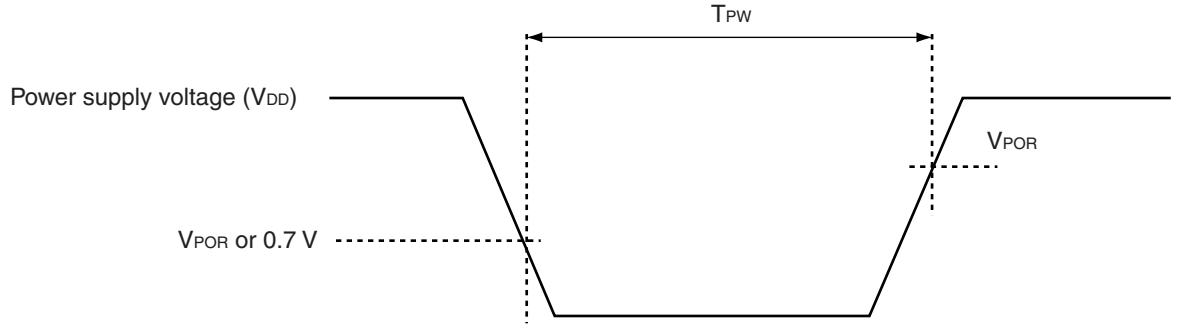
| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Temperature sensor output voltage | $V_{TMPS25}$ | Setting ADS register = 80H, $T_A$ = +25°C | | 1.05 | | V |
| Internal reference voltage | $V_{BGR}$ | Setting ADS register = 81H | 1.38 | 1.45 | 1.5 | V |
| Temperature coefficient | $F_{VTMPS}$ | Temperature sensor output voltage that depends on the temperature | | –3.6 | | mV/°C |
| Operation stabilization wait time | $t_{AMP}$ | | 10 | | | μs |

### 30.6.3 POR circuit characteristics

**($T_A$ = –40 to +105°C, $V_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Detection voltage | $V_{POR}$ | Power supply rise time | 1.45 | 1.51 | 1.57 | V |
| | $V_{PDR}$ | Power supply fall time | 1.44 | 1.50 | 1.56 | V |
| Minimum pulse width[Note] | $T_{PW}$ | | 300 | | | μs |

**Note** This is the time required for the POR circuit to execute a reset when $V_{DD}$ falls below $V_{PDR}$. When the microcontroller enters STOP mode or if the main system clock ($f_{MAIN}$) has been stopped by setting bit 0 (HIOSTOP) and bit 7 (MSTOP) of the clock operation status control register (CSC), this is the time required for the POR circuit to execute a reset before $V_{DD}$ rises to $V_{POR}$ after having fallen below 0.7 V.



### 30.6.4 LVD circuit characteristics

**LVD Detection Voltage of Reset Mode and Interrupt Mode**

**($T_A$ = −40 to +105°C, $V_{PDR} \le V_{DD} \le 3.6$ V, $V_{SS}$ = 0 V)**

| Parameter | | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| Detection voltage | Supply voltage level | $V_{LVD2}$ | Power supply rise time | 3.01 | 3.13 | 3.25 | V |
| | | | Power supply fall time | 2.94 | 3.06 | 3.18 | V |
| | | $V_{LVD3}$ | Power supply rise time | 2.90 | 3.02 | 3.14 | V |
| | | | Power supply fall time | 2.85 | 2.96 | 3.07 | V |
| | | $V_{LVD4}$ | Power supply rise time | 2.81 | 2.92 | 3.03 | V |
| | | | Power supply fall time | 2.75 | 2.86 | 2.97 | V |
| | | $V_{LVD5}$ | Power supply rise time | 2.70 | 2.81 | 2.92 | V |
| | | | Power supply fall time | 2.64 | 2.75 | 2.86 | V |
| | | $V_{LVD6}$ | Power supply rise time | 2.61 | 2.71 | 2.81 | V |
| | | | Power supply fall time | 2.55 | 2.65 | 2.75 | V |
| | | $V_{LVD7}$ | Power supply rise time | 2.51 | 2.61 | 2.71 | V |
| | | | Power supply fall time | 2.45 | 2.55 | 2.65 | V |
| Minimum pulse width | | $t_{LW}$ | | 300 | | | μs |
| Detection delay time | | | | | | 300 | μs |

**Remark** $V_{LVD\,(n-1)} > V_{LVDn}$: n = 3 to 7

**LVD Detection Voltage of Interrupt & Reset Mode**

**($T_A$ = −40 to +105°C, $V_{PDR} \le V_{DD} \le 3.6$ V, $V_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|---|
| Interrupt & reset mode | $V_{LVD5}$ | VPOC2, VPOC1, VPOC0 = 0, 1, 1, falling reset voltage | | | 2.64 | 2.75 | 2.86 | V |
| | $V_{LVD4}$ | | LVIS1, LVIS0 = 1, 0 | Rising release reset voltage | 2.81 | 2.92 | 3.03 | V |
| | | | | Falling interrupt voltage | 2.75 | 2.86 | 2.97 | V |
| | $V_{LVD3}$ | | LVIS1, LVIS0 = 0, 1 | Rising release reset voltage | 2.90 | 3.02 | 3.14 | V |
| | | | | Falling interrupt voltage | 2.85 | 2.96 | 3.07 | V |

**Caution Set the detection voltage ($V_{LVD}$) to be within the operating voltage range. The operating voltage range depends on the setting of the user option byte (000C2H/010C2H). The following shows the operating voltage range.**

**HS (high-speed main) mode: $V_{DD}$ = 2.7 to 3.6 V@1 MHz to 32 MHz**

**$V_{DD}$ = 2.4 to 3.6 V@1 MHz to 16 MHz**

### 30.6.5 Supply voltage rise slope characteristics

**($T_A$ = −40 to +105°C, $V_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Supply voltage rise | $SV_{DD}$ | | | | 54 | V/ms |

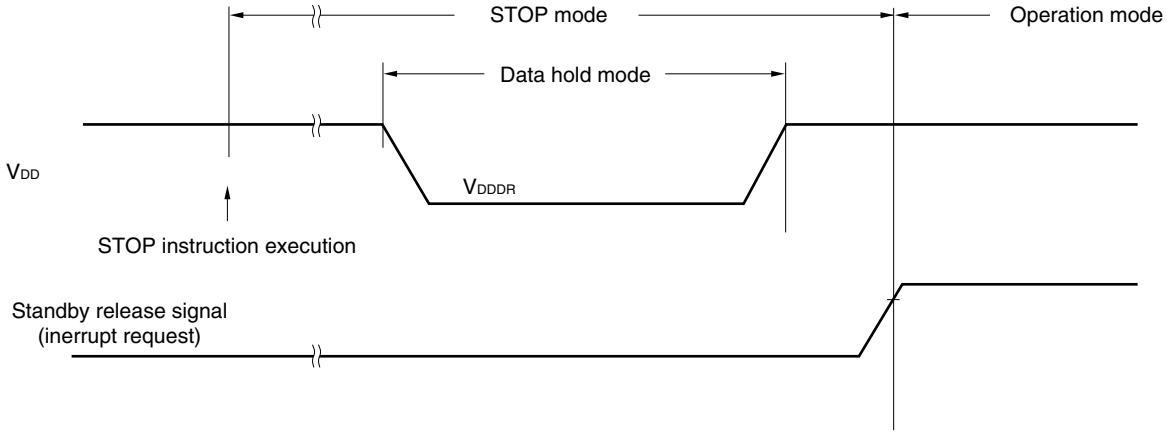
**Caution Be sure to maintain the internal reset state until $V_{DD}$ reaches the operating voltage range specified in 30.4 AC Characteristics, by using the LVD circuit or external reset pin.**

## 30.7 Data Memory STOP Mode Low Supply Voltage Data Retention Characteristics

<R> **($T_A$ = −40 to +105°C, $V_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Data retention supply voltage | $V_{DDDR}$ | | 1.44[Note] | | 3.6 | V |

**Note** The value depends on the POR detection voltage. When the voltage drops, the data is retained before a POR reset is effected, but data is not retained when a POR reset is effected.




## 30.8 Flash Memory Programming Characteristics

**($T_A$ = −40 to +105°C, 2.4 V ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = 0 V)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| CPU/peripheral hardware clock frequency | $f_{CLK}$ | 2.4 V ≤ $V_{DD}$ ≤ 3.6 V | | 1 | | 32 | MHz |
| Number of code flash rewrites[Notes 1, 2, 3] | $C_{erwr}$ | Retained for 20 years | $T_A$ = 85°C | 1,000 | | | Times |
| Number of data flash rewrites[Notes 1, 2, 3] | | Retained for 1 years | $T_A$ = 25°C | | 1,000,000 | | |
| | | Retained for 5 years | $T_A$ = 85°C | 100,000 | | | |
| | | Retained for 20 years | $T_A$ = 85°C | 10,000 | | | |

**Notes 1.** 1 erase + 1 write after the erase is regarded as 1 rewrite.
The retaining years are until next rewrite after the rewrite.

**2.** When using flash memory programmer and Renesas Electronics self programming library

**3.** These are the characteristics of the flash memory and the results obtained from reliability testing by Renesas Electronics Corporation.

## 30.9 Dedicated Flash Memory Programmer Communication (UART)

**($T_A$ = −40 to +105°C, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

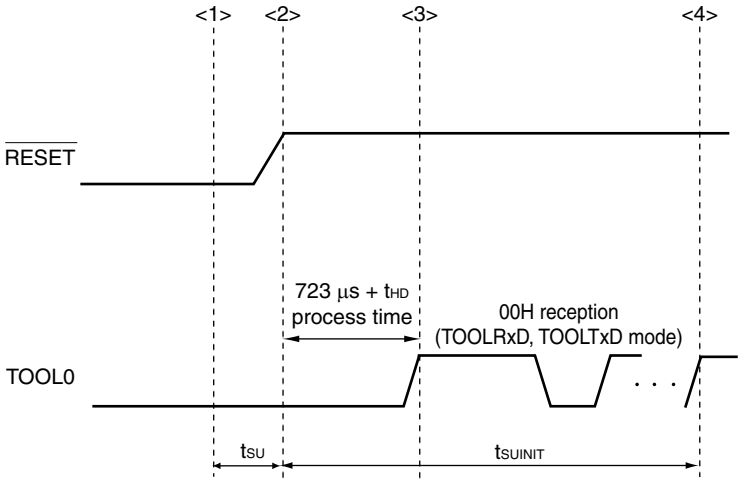
| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|
| Transfer rate | | During flash memory programming | 115.2 k | | 1 M | bps |

## 30.10 Timing Specs for Switching Flash Memory Programming Modes

**($T_A$ = −40 to +105°C, 2.4 V ≤ $EV_{DD0}$ ≤ $V_{DD}$ ≤ 3.6 V, $V_{SS}$ = $EV_{SS0}$ = 0 V)**

| | Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|---|---|---|---|---|---|
| | How long from when an external reset ends until the initial communication settings are specified | $t_{SUINIT}$ | POR and LVD reset must end before the external reset ends. | | | 100 | ms |
| | How long from when the TOOL0 pin is placed at the low level until a external reset ends | $t_{SU}$ | POR and LVD reset must end before the external reset ends. | 10 | | | μs |
| <R> | How long the TOOL0 pin must be kept at the low level after an external reset ends (except flash firmware processing time) | $t_{HD}$ | POR and LVD reset must end before the external reset ends. | 1 | | | ms |

<R>




<1> The low level is input to the TOOL0 pin.

<2> The pins reset ends (POR and LVD reset must end before the external reset ends.).

<3> The TOOL0 pin is set to the high level.

<4> Setting of the flash memory programming mode by UART reception and complete the baud rate setting.

**Remark** $t_{SUINIT}$: The segment shows that it is necessary to finish specifying the initial communication settings within 100 ms from when the resets end.

$t_{SU}$: How long from when the TOOL0 pin is placed at the low level until a external reset ends

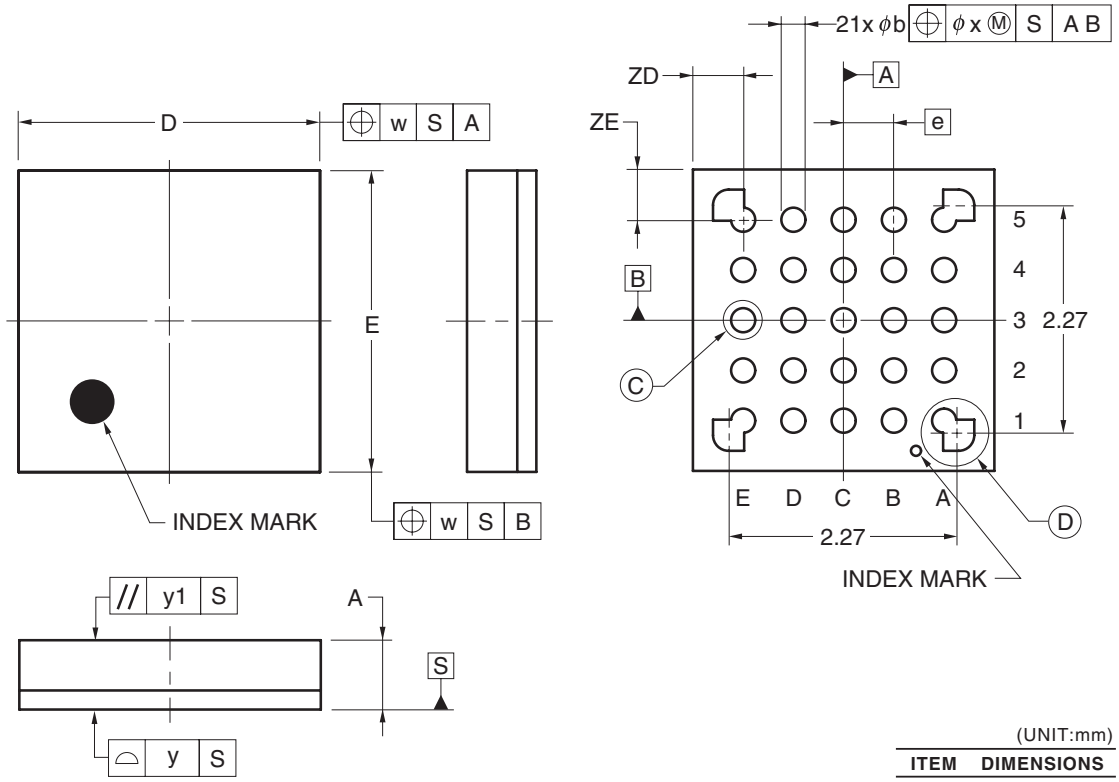
<R> $t_{HD}$: How long to keep the TOOL0 pin at the low level from when the external resets end (except flash firmware processing time)

# CHAPTER 31 PACKAGE DRAWINGS

## <R> 31.1 25-pin Products

R5F10E8AALA, R5F10E8CALA, R5F10E8DALA, R5F10E8EALA

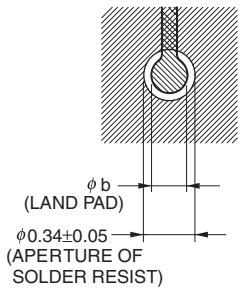
| JEITA Package Code | RENESAS Code | Previous Code | MASS (TYP.) [g] |
|---|---|---|---|
| P-WFLGA25-3x3-0.50 | PWLG0025KA-A | P25FC-50-2N2-2 | 0.01 |



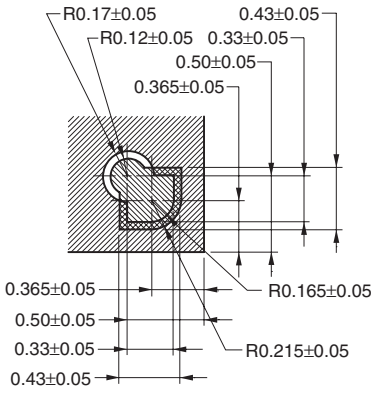

(UNIT:mm)

| ITEM | DIMENSIONS |
|---|---|
| D | 3.00±0.10 |
| E | 3.00±0.10 |
| w | 0.20 |
| e | 0.50 |
| A | 0.69±0.07 |
| b | 0.24±0.05 |
| x | 0.05 |
| y | 0.08 |
| y1 | 0.20 |
| ZD | 0.50 |
| ZE | 0.50 |

DETAIL OF Ⓒ PART



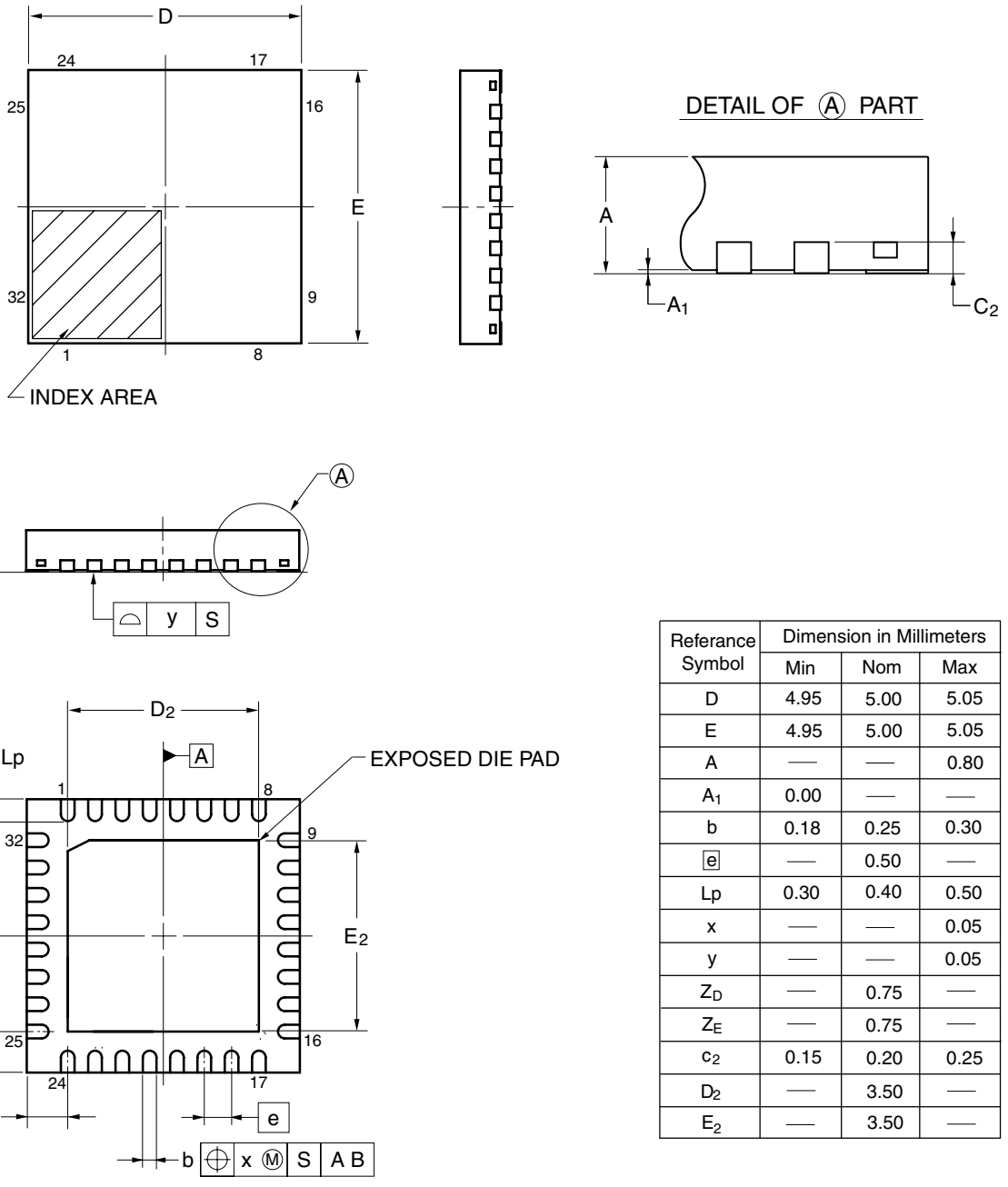

DETAIL OF Ⓓ PART



## <R> 31.2 32-pin Products

R5F10EBAANA, R5F10EBCANA, R5F10EBDANA, R5F10EBEANA
R5F10EBAGNA, R5F10EBCGNA, R5F10EBDGNA, R5F10EBEGNA

| JEITA Package code | RENESAS code | Previous code | MASS(TYP.)[g] |
|---|---|---|---|
| P-HWQFN32-5x5-0.50 | PWQN0032KB-A | P32K8-50-3B4-5 | 0.06 |

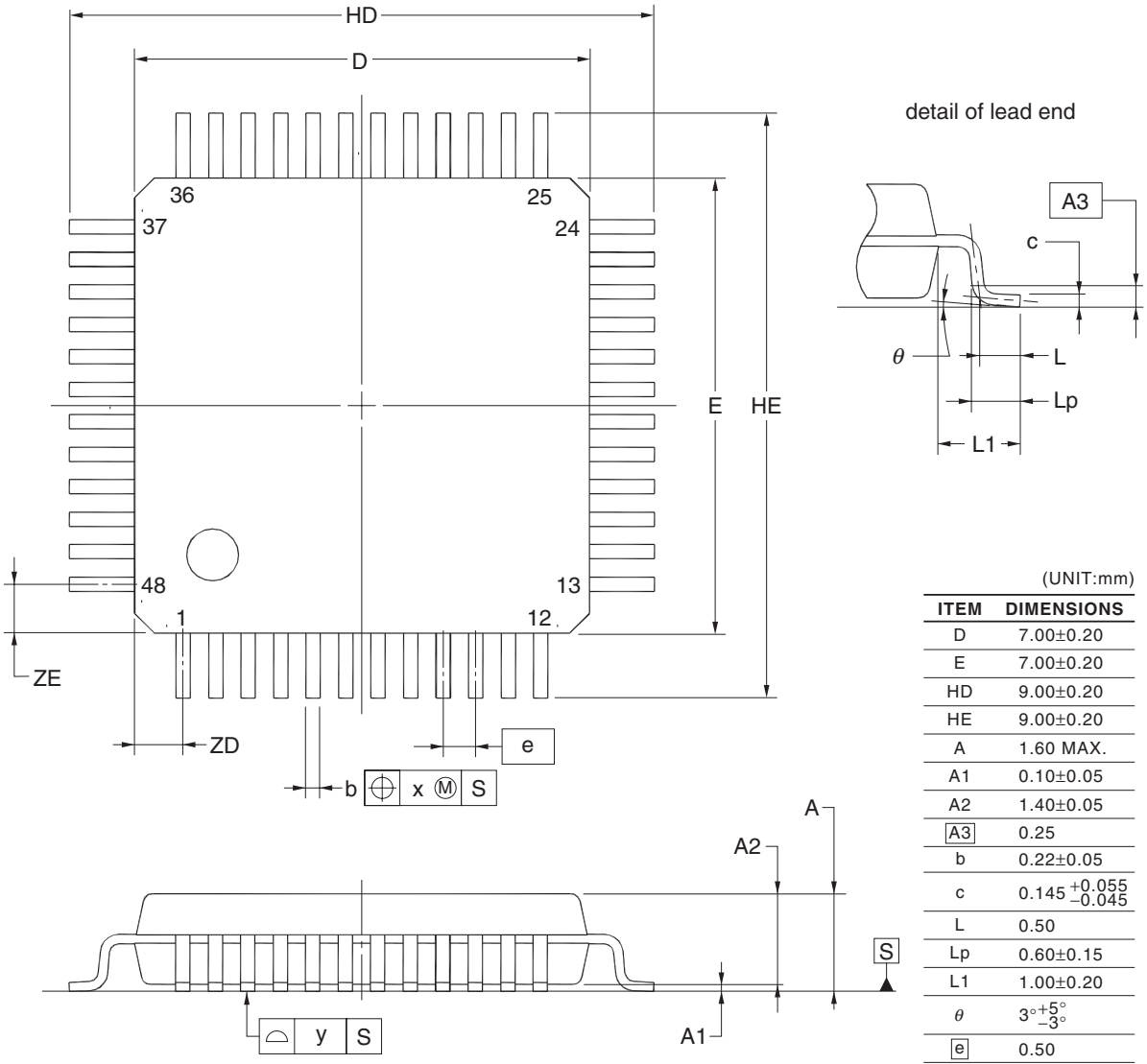



| Referance Symbol | Dimension in Millimeters | | |
|---|---|---|---|
| | Min | Nom | Max |
| D | 4.95 | 5.00 | 5.05 |
| E | 4.95 | 5.00 | 5.05 |
| A | — | — | 0.80 |
| $A_1$ | 0.00 | — | — |
| b | 0.18 | 0.25 | 0.30 |
| e | — | 0.50 | — |
| Lp | 0.30 | 0.40 | 0.50 |
| x | — | — | 0.05 |
| y | — | — | 0.05 |
| $Z_D$ | — | 0.75 | — |
| $Z_E$ | — | 0.75 | — |
| $c_2$ | 0.15 | 0.20 | 0.25 |
| $D_2$ | — | 3.50 | — |
| $E_2$ | — | 3.50 | — |

## <R> 31.3 48-pin Products

R5F10EGAAFB, R5F10EGCAFB, R5F10EGDAFB, R5F10EGEAFB
R5F10EGAGFB, R5F10EGCGFB, R5F10EGDGFB, R5F10EGEGFB

| JEITA Package Code | RENESAS Code | Previous Code | MASS (TYP.) [g] |
|---|---|---|---|
| P-LFQFP48-7x7-0.50 | PLQP0048KF-A | P48GA-50-8EU-1 | 0.16 |



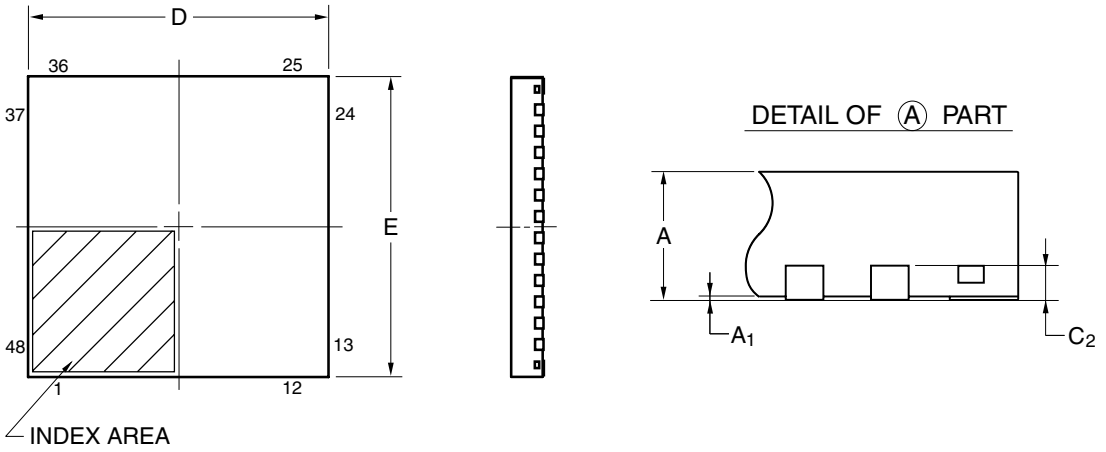

(UNIT:mm)

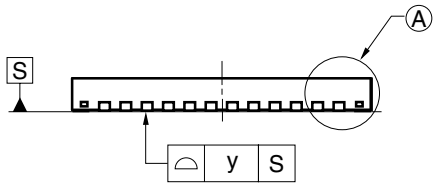
| ITEM | DIMENSIONS |
|---|---|
| D | 7.00±0.20 |
| E | 7.00±0.20 |
| HD | 9.00±0.20 |
| HE | 9.00±0.20 |
| A | 1.60 MAX. |
| A1 | 0.10±0.05 |
| A2 | 1.40±0.05 |
| A3 | 0.25 |
| b | 0.22±0.05 |
| c | $0.145^{+0.055}_{-0.045}$ |
| L | 0.50 |
| Lp | 0.60±0.15 |
| L1 | 1.00±0.20 |
| θ | $3^{\circ}{}^{+5^{\circ}}_{-3^{\circ}}$ |
| e | 0.50 |
| x | 0.08 |
| y | 0.08 |
| ZD | 0.75 |
| ZE | 0.75 |

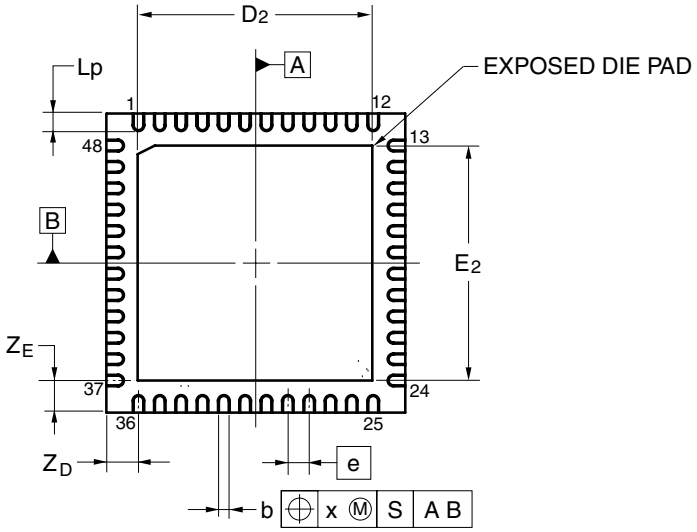
**NOTE**
Each lead centerline is located within 0.08 mm of its true position at maximum material condition.

R5F10EGAANA, R5F10EGCANA, R5F10EGDANA, R5F10EGEANA
R5F10EGAGNA, R5F10EGCGNA, R5F10EGDGNA, R5F10EGEGNA

| JEITA Package code | RENESAS code | Previous code | MASS(TYP.)[g] |
|---|---|---|---|
| P-HWQFN48-7x7-0.50 | PWQN0040KB-A | 48PJN-A<br>P40K8-50-5B4-6 | 0.13 |



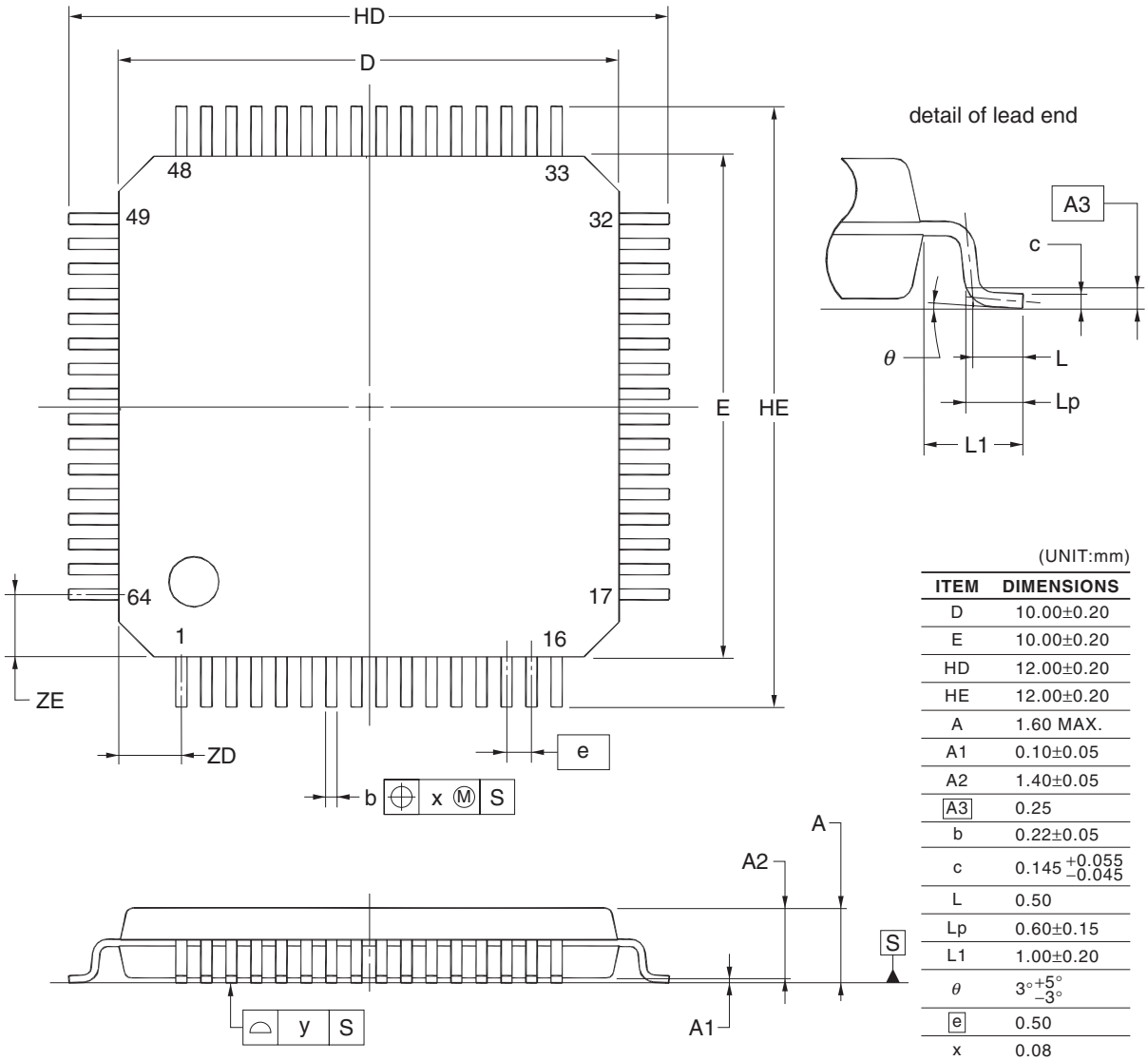




| Referance Symbol | Dimension in Millimeters | | |
|---|---|---|---|
| | Min | Nom | Max |
| D | 6.95 | 7.00 | 7.05 |
| E | 6.95 | 7.00 | 7.05 |
| A | — | — | 0.80 |
| $A_1$ | 0.00 | — | — |
| b | 0.18 | 0.25 | 0.30 |
| e | — | 0.50 | — |
| Lp | 0.30 | 0.40 | 0.50 |
| x | — | — | 0.05 |
| y | — | — | 0.05 |
| $Z_D$ | — | 0.75 | — |
| $Z_E$ | — | 0.75 | — |
| $c_2$ | 0.15 | 0.20 | 0.25 |
| $D_2$ | — | 5.50 | — |
| $E_2$ | — | 5.50 | — |

## <R> 31.4 64-pin Products

R5F10ELCAFB, R5F10ELDAFB, R5F10ELEAFB
R5F10ELCGFB, R5F10ELDGFB, R5F10ELEGFB

| JEITA Package Code | RENESAS Code | Previous Code | MASS (TYP.) [g] |
|---|---|---|---|
| P-LFQFP64-10x10-0.50 | PLQP0064KF-A | P64GB-50-UEU-2 | 0.35 |



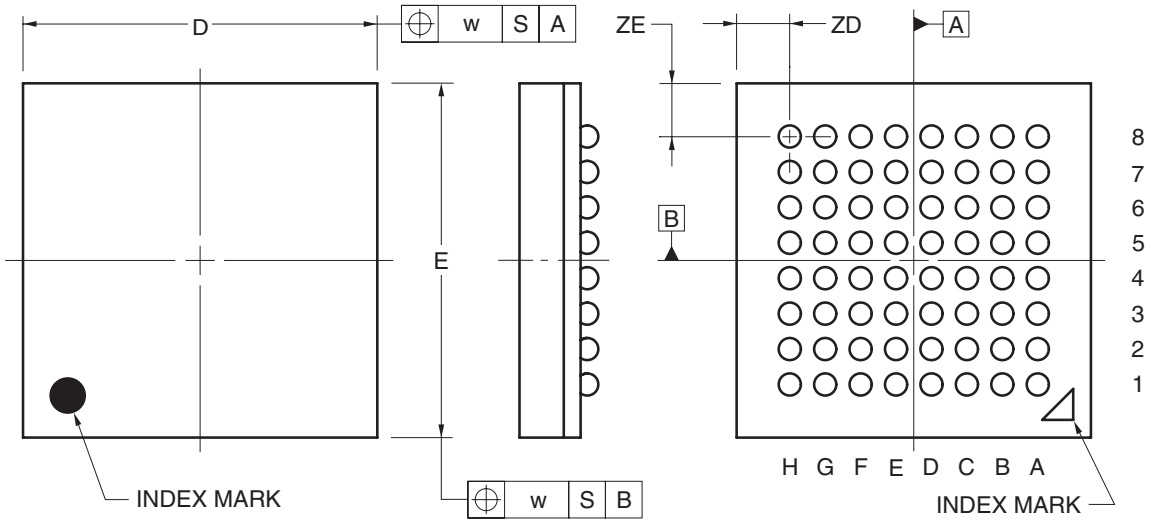

(UNIT:mm)

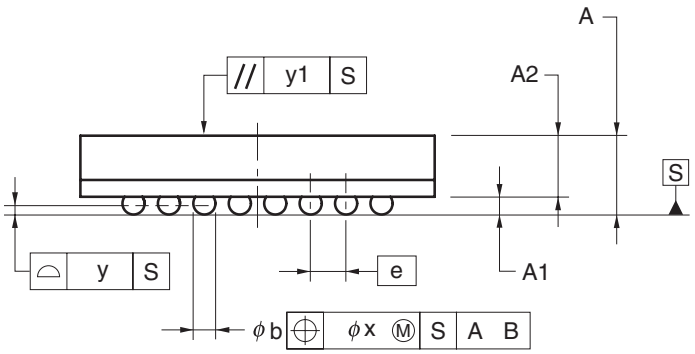
| ITEM | DIMENSIONS |
|---|---|
| D | 10.00±0.20 |
| E | 10.00±0.20 |
| HD | 12.00±0.20 |
| HE | 12.00±0.20 |
| A | 1.60 MAX. |
| A1 | 0.10±0.05 |
| A2 | 1.40±0.05 |
| A3 | 0.25 |
| b | 0.22±0.05 |
| c | $0.145^{+0.055}_{-0.045}$ |
| L | 0.50 |
| Lp | 0.60±0.15 |
| L1 | 1.00±0.20 |
| θ | $3^{\circ}{}^{+5^{\circ}}_{-3^{\circ}}$ |
| e | 0.50 |
| x | 0.08 |
| y | 0.08 |
| ZD | 1.25 |
| ZE | 1.25 |

**NOTE**
Each lead centerline is located within 0.08 mm of its true position at maximum material condition.

R5F10ELCABG, R5F10ELDABG, R5F10ELEABG

| JEITA Package Code | RENESAS Code | Previous Code | MASS (TYP.) [g] |
|---|---|---|---|
| P-VFBGA64-4x4-0.40 | PVBG0064LA-A | P64F1-40-AA2-2 | 0.03 |




(UNIT:mm)

| ITEM | DIMENSIONS |
|---|---|
| D | 4.00±0.10 |
| E | 4.00±0.10 |
| w | 0.15 |
| A | 0.89±0.10 |
| A1 | 0.20±0.05 |
| A2 | 0.69 |
| e | 0.40 |
| b | 0.25±0.05 |
| x | 0.05 |
| y | 0.08 |
| y1 | 0.20 |
| ZD | 0.60 |
| ZE | 0.60 |

# APPENDIX A REVISION HISTORY

## A.1 Major Revisions in This Edition

(1/11)

| Page | Description | Classification |
|---|---|---|
| R01UH0305EJ0110 → R01UH0305EJ0200 | | |
| **CHAPTER 1 OUTLINE** | | |
| p.1 | Modification of **1.1 Features** | (b) |
| p.5 | Modification of **Table 1-1. List of Ordering Part Numbers** | (b) |
| p.7 | Modification of **Remark 3** | (b) |
| p.17 | Modification of **1.6 Outline of Functions** | (b) |
| **CHAPTER 2 PIN FUNCTIONS** | | |
| p.20, 21 | Modification of **2.1.1 25-pin products** | (b) |
| p.22, 23 | Modification of **2.1.2 32-pin products** | (b) |
| p.24, 25 | Modification of **2.1.3 48-pin products** | (b) |
| p.26 to 28 | Modification of **2.1.4 64-pin products** | (b) |
| p.33 | Modification of **2.2.2 Explanation of function** | |
| p.35 | Modification of **2.3 Pin I/O Circuits and Recommended Connection of Unused Pins** | (b) |
| p.35 | Modification of **Table 2-3. Connection of Unused Pins** | (b) |
| p.36 to 46 | Modification of **Figures 2-1** to **2-13** in **2.4 Block Diagrams of Pins** | (b) |
| **CHAPTER 3 CPU ARCHITECTURE** | | |
| p.57 | Modification of **3.1.2 Mirror area** | (c) |
| p.61 | Modification of **3.1.6 Data memory addressing** | (c) |
| p.63 | Modification of figure and **Cautions 3** and **4** in **Figure 3-9** | (c) |
| p.65 | Modification of **Figures 3-11** and **3-12** | (c) |
| p.70 | Modification of **Note 1** in **Table 3-5** | (d) |
| p.74 | Modification of description and **Note** in **Table 3-6. Extended SFR (2nd SFR) List** | (d) |
| p.76 | Modification of **Table 3-6. Extended SFR (2nd SFR) List** | (a) |
| **CHAPTER 4 PORT FUNCTIONS** | | |
| p.97 | Modification of **Table 4-1. Port Configuration** | (c) |
| p.98 | Modification of **4.2.1 Port 0** and **4.2.2 Port 1** | (c) |
| p.99 | Modification of **4.2.3 Port 2** and **4.2.4 Port 3** | (c) |
| p.100 | Modification of **4.2.5 Port 4** and **4.2.6 Port 5** | (c) |
| p.101 | Modification of **4.2.8 Port 7** and **4.2.9 Port 12** | (c) |
| p.102 | Modification of **4.2.12 Port 15** | (c) |
| p.105 | Modification of **4.3.1 Port mode registers (PMxx)** | (c) |
| p.106 | Modification of **4.3.2 Port registers (Pxx)** | (c) |
| p.107 | Modification of **4.3.3 Pull-up resistor option registers (PUxx)** | (c) |
| p.108 | Modification of **4.3.4 Port input mode registers (PIMxx)** | (c) |
| p.109 | Modification of **4.3.5 Port output mode registers (POMxx)** | (c) |

**Remark** "Classification" in the above table classifies revisions as follows.
(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note, (d): Addition/change of package, part number, or management division, (e): Addition/change of related documents

(2/11)

| Page | Description | Classification |
|---|---|---|
| p.110 | Modification of **4.3.6 Port mode control registers (PMCxx)** | (c) |
| p.111 | Modification of **4.3.7 A/D port configuration register (ADPC)** | (c) |
| p.112 | Modification of **4.3.8 Peripheral I/O redirection register (PIOR)** | (c) |
| p.113 | Modification of **4.3.9 Global digital input disable register (GDIDIS)** | (c) |
| p.114 | Modification of **4.3.10 Global analog input disable register (GAIDIS)** | (c) |
| p.116 | Modification of **4.4.4 Handling different potential (1.8 V or 2.5 V) by using $EV_{DD} \leq V_{DD}$** | (c) |
| p.116, 117 | Modification of **4.4.5 Handling different potential (1.8 V or 2.5 V) by using I/O buffers** | (c) |
| p.118 | Modification of **4.5 Register Settings When Using Alternate Function** | (c) |
| p.119 | Modification of **4.5.2 Register settings for alternate function whose output function is not used** | (c) |
| p.120 | Modification of **4.5.3 Register setting examples for used port and alternate functions** | (c) |
| p.121 to 136 | Modification of **Table 4-6. Setting Examples of Registers and Output Latches When Using Alternate Function** | (c) |
| p.138 | Modification of **4.6.2 Notes on specifying the pin settings** | (c) |
| **CHAPTER 5 CLOCK GENERATOR** | | |
| p.140 | Modification of **(3) Low-speed on-chip oscillator clock (Low-speed on-chip oscillator)** | (c) |
| p.141 | Modification of **Table 5-1. Configuration of Clock Generator** | (c) |
| p.142 | Modification of **Figure 5-1. Block Diagram of Clock Generator** | (c) |
| p.143 | Modification of **5.3 Registers Controlling Clock Generator** | (c) |
| p.145 | Modification of **Caution 7** | (c) |
| p.149 | Modification of **Figure 5-5. Format of Oscillation Stabilization Time Counter Status Register (OSTC)** | (c) |
| p.150 | Modification of **5.3.5 Oscillation stabilization time select register (OSTS)** | (c) |
| p.151 | Modification of **Figure 5-6. Format of Oscillation Stabilization Time Select Register (OSTS)** | (c) |
| p.155 | Modification of **5.3.7 Subsystem clock supply mode control register (OSMC)** | (c) |
| p.156 | Modification of **5.3.8 High-speed on-chip oscillator frequency select register (HOCODIV)** | (c) |
| p.159 | Modification of **Caution** in **Figure 5-12. Example of External Circuit of XT1 Oscillator** | (c) |
| p.162 | Modification of **5.4.4 Low-speed on-chip oscillator** | (c) |
| p.167 | Modification of **5.6.3 Example of setting XT1 oscillation clock** | (c) |
| p.170 to 173 | Modification of **Table 5-3. CPU Clock Transition and SFR Register Setting Examples** | (c) |
| p.174 | Modification of **5.6.5 Condition before changing CPU clock and processing after changing CPU clock** | (c) |
| p.178 | Modification of **Figure 5-16. External Oscillation Circuit Example** | (c) |
| p.181 | Modification of **(1) X1 oscillation** and **(2) XT1 oscillation** | (c) |
| **CHAPTER 6 TIMER ARRAY UNIT** | | |
| p.185 | Modification of **6.1.2 Simultaneous channel operation function** | (c) |
| p.188 | Modification of **Table 6-1. Configuration of Timer Array Unit** | (c) |
| p.191 | Modification of **Figure 6-3. Internal Block Diagram of Channels 1 and 3 of Timer Array Unit 0** | (c) |
| p.197 | Modification of **Figure 6-9. Format of Peripheral Enable Register 0 (PER0)** | (c) |

**Remark** "Classification" in the above table classifies revisions as follows.
(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note, (d): Addition/change of package, part number, or management division, (e): Addition/change of related documents

(3/11)

| Page | Description | Classification |
|---|---|---|
| p.198 | Modification of **6.3.2 Timer clock select register m (TPSm)** | (c) |
| p.199 | Modification of **Caution** in **Figure 6-10. Format of Timer Clock Select register m (TPSm)** | (c) |
| p.215 | Modification of **Figure 6-21. Format of Input Switch Control Register (ISC)** | (c) |
| p.216 | Modification of **6.3.14 Noise filter enable register 1 (NFEN1)** | (c) |
| p.218 | Modification of **6.3.15 Registers controlling port functions of pins to be used for timer I/O** | (c) |
| p.226 | Modification of **Remark** in **Figure 6-26. Operation Timing (In Event Counter Mode)** | (c) |
| p.227 | Modification of **Remark 1** | (c) |
| p.245 | Modification of **Figure 6-44. Operation Procedure of Interval Timer/Square Wave Output Function** | (c) |
| p.251 | Modification of **Figure 6-48. Operation Procedure When External Event Counter Function Is Used** | (c) |
| p.254 | Modification of **Figure 6-51. Example of Set Contents of Registers During Operation as Frequency Divider** | (c) |
| p.255 | Modification of **Figure 6-52. Operation Procedure When Frequency Divider Function Is Used** | (c) |
| p.256 | Modification of **6.8.4 Operation as input pulse interval measurement** | (c) |
| p.259 | Modification of F**igure 6-56. Operation Procedure When Input Pulse Interval Measurement Function Is Used** | (c) |
| p.261 | Modification of **Figure 6-57. Block Diagram of Operation as Input Signal High-/Low-Level Width Measurement** | (c) |
| p.264 | Modification of **Figure 6-60. Operation Procedure When Input Signal High-/Low-Level Width Measurement Function Is Used** | (c) |
| p.268 | Modification of **Figure 6-64. Operation Procedure When Delay Counter Function Is Used** | (c) |
| p.272 | Modification of **Figure 6-67. Example of Set Contents of Registers When One-Shot Pulse Output Function Is Used (Master Channel)** | (c) |
| p.274, 275 | Modification of **Figure 6-69. Operation Procedure of One-Shot Pulse Output Function** | (c) |
| p.279 | Modification of **Figure 6-72. Example of Set Contents of Registers When PWM Function (Master Channel) Is Used** | (c) |
| p.287 | Modification of **Figure 6-77. Example of Set Contents of Registers When Multiple PWM Output Function (Master Channel) Is Used** | (c) |
| p.291 | Modification of **6.10.1 Cautions When Using Timer output** | (c) |
| **CHAPTER 7 REAL-TIME CLOCK** | | |
| p.292 | Modification of **7.1 Functions of Real-time Clock** | (c) |
| p.294 | Modification of **Figure 7-1. Block Diagram of Real-time Clock** | (c) |
| p.295 | Modification of **7.3 Registers Controlling Real-time Clock** | (c) |
| p.296 | Modification of **7.3.1 Peripheral enable register 0 (PER0)** | (c) |
| p.298 | Modification of **7.3.3 Real-time clock control register 0 (RTCC0)** | (c) |
| p.299, 300 | Modification of **7.3.4 Real-time clock control register 1 (RTCC1)** | (c) |
| p.301 | Modification of **7.3.5 Second count register (SEC)** | (c) |
| p.302 | Modification of **7.3.7 Hour count register (HOUR)** | (c) |
| p.304 | Modification of **7.3.8 Day count register (DAY)** | (c) |
| p.305 | Modification of **7.3.9 Week count register (WEEK)** | (c) |

**Remark** "Classification" in the above table classifies revisions as follows.
(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note, (d): Addition/change of package, part number, or management division, (e): Addition/change of related documents

(4/11)

| Page | Description | Classification |
|---|---|---|
| p.306 | Modification of **7.3.10 Month count register (MONTH)** | (c) |
| p.309 | Modification of **7.3.16 Port mode register 3 (PM3)** | (c) |
| p.309 | Modification of **7.3.17 Port register 3 (P3)** | (c) |
| p.315 | Modification of **7.4.5 1 Hz output of real-time clock** | (c) |
| p.317 | Modification of **Correction example 1** in **7.4.6 Example of watch error correction of real-time clock** | (c) |
| p.318 | Modification of **Figure 7-25. Correction Operation when (DEV, F6, F5, F4, F3, F2, F1, F0) = (0, 0, 1, 0, 1, 1, 0, 0)** | (c) |
| p.320 | Modification of **Figure 7-26. Correction Operation when (DEV, F6, F5, F4, F3, F2, F1, F0) = (1, 1, 1, 0, 1, 1, 1, 0)** | (c) |
| **CHAPTER 8 12-BIT INTERVAL TIMER** | | |
| p.321 | Modification of **8.2 Configuration of 12-bit Interval Timer** | (c) |
| p.322 | Modification of **8.3.1 Peripheral enable register 0 (PER0)** | (c) |
| p.325 | Modification of **Figure 8-5. 12-bit Interval Timer Operation Timing** | (c) |
| **CHAPTER 9 CLOCK OUTPUT/BUZZER OUTPUT CONTROLLER** | | |
| p.327 | Modification of **9.1 Functions of Clock Output/Buzzer Output Controller** | (c) |
| p.330 | Modification of **Figure 9-2. Format of Clock Output Select Register n (CKSn)** | (c) |
| p.331 | Modification of **9.3.2 Registers controlling port functions of pins to be used for clock or buzzer output** | (c) |
| p.332 | Modification of **9.4 Operations of Clock Output/Buzzer Output Controller** | (c) |
| **CHAPTER 10 WATCHDOG TIMER** | | |
| p.333 | Modification of **10.1 Functions of Watchdog Timer** | (c) |
| p.334 | Modification of **10.2 Configuration of Watchdog Timer** | (c) |
| p.336 | Modification of **10.4.1 Controlling operation of watchdog timer** | (c) |
| p.337 | Modification of **Table 10-3. Setting of Overflow Time of Watchdog Timer** | (c) |
| **CHAPTER 11 A/D CONVERTER** | | |
| p.340, 341 | Modification of **11.1 Function of A/D Converter** | (c) |
| p.346 | Modification of **Figure 11-2. Format of Peripheral Enable Register 0 (PER0)** | (c) |
| p.347 | Modification of **Figure 11-3. Format of A/D Converter Mode Register 0 (ADM0)** | (c) |
| p.350 to 357 | Modification of **Table 11-3. A/D Conversion Time Selection** | (c) |
| p.359 | Modification of **Figure 11-6. Format of A/D Converter Mode Register 1 (ADM1)** | (c) |
| p.361 | Modification of **Figure 11-7. Format of A/D Converter Mode Register 2 (ADM2) (2/2)** | (c) |
| p.362 | Modification of **11.3.5 12-bit A/D conversion result register (ADCR)** | (c) |
| p.365 | Modification of **Figure 11-11. Format of Analog Input Channel Specification Register (ADS) (2/2)** | (c) |
| p.367 | Modification of **11.3.10 A/D test register (ADTES)** and addition of **Caution** | (c) |
| p.368 | Modification of **11.3.11 Registers controlling port function of analog input pins** | (c) |
| p.370 | Modification of **Figure 11-15. Conversion Operation of A/D Converter (Software Trigger Mode)** | (c) |
| p.385 to 389, 393 | Modification of **Figures 11-29**, **11-30**, **11-31**, **11-32**, **11-33**, and **11-37** | (c) |
| p.389 | Modification of **Figure 11-33. Setting up Test Mode** | (c) |

**Remark** "Classification" in the above table classifies revisions as follows.
(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note, (d): Addition/change of package, part number, or management division, (e): Addition/change of related documents

(5/11)

| Page | Description | Classification |
|---|---|---|
| p.393 | Modification of **Figure 11-37. Flowchart for Setting up SNOOZE Mode** | (c) |
| p.396 | Modification of **11.10 Cautions for A/D Converter** | (c) |
| **CHAPTER 12 SERIAL ARRAY UNIT** | | |
| p.405 | Modification of **Figure 12-1. Block Diagram of Serial Array Unit 0** | (c) |
| p.406 | Modification of **Figure 12-2. Block Diagram of Serial Array Unit 1** | (c) |
| p.407 | Modification of **12.2.1 Shift register** | (c) |
| p.408 | Modification of **Figure 12-3. Format of Serial Data Register mn (SDRmn) (mn = 00, 01, 02, 03, 10, 11)** | (c) |
| p.410 | Modification of **Figure 12-4. Format of Peripheral Enable Register 0 (PER0)** | (c) |
| p.411 | Modification of **Figure 12-5. Format of Serial Clock Select Register m (SPSm)** | (c) |
| p.412, 413 | Modification of **Figure 12-6. Format of Serial Mode Register mn (SMRmn)** | (c) |
| p.414, 415 | Modification of **Figure 12-7. Format of Serial Communication Operation Setting Register mn (SCRmn)** | (c) |
| p.416, 417 | Modification of **12.3.5 Higher 7 bits of the serial data register mn (SDRmn)** | (c) |
| p.418 | Modification of **12.3.6 Serial flag clear trigger register mn (SIRmn)** | (c) |
| p.419, 420 | Modification of **12.3.7 Serial status register mn (SSRmn)** | (c) |
| p.425 | Modification of **12.3.12 Serial output register m (SOm)** | (c) |
| p.427 | Modification of **Figure 12-17. Examples of Reverse Transmit Data** | (c) |
| p.428 | Modification of **12.3.14 Serial standby control register 0 (SSC0)** | (c) |
| p.429 | Modification of **12.3.15 Input switch control register (ISC)** | (c) |
| p.430 | Modification of **12.3.16 Noise filter enable register 0 (NFEN0)** | (c) |
| p.434 | Modification of **12.5 Operation of 3-Wire Serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) Communication** | (c) |
| p.437 | Modification of **12.5.1 Master transmission** | (c) |
| p.441 | Modification of **Figure 12-26. Procedure for Stopping Master Transmission** | (c) |
| p.443 | Modification of **Figure 12-28. Timing Chart of Master Transmission (in Single-Transmission Mode) (Type 1: DAPmn = 0, CKPmn = 0)** | (c) |
| p.444 | Modification of **Figure 12-29. Flowchart of Master Transmission (in Single-Transmission Mode)** | (c) |
| p.445 | Modification of **Figure 12-30. Timing Chart of Master Transmission (in Continuous Transmission Mode) (Type 1: DAPmn = 0, CKPmn = 0)** | (c) |
| p.446 | Modification of **Figure 12-31. Flowchart of Master Transmission (in Continuous Transmission Mode)** | (c) |
| p.447 | Modification of **12.5.2 Master reception** | (c) |
| p.453 | Modification of **Figure 12-36. Timing Chart of Master Reception (in Single-Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)** | (c) |
| p.454 | Modification of **Figure 12-37. Flowchart of Master Reception (in Single-Reception Mode)** | (c) |
| p.455 | Modification of **Figure 12-38. Timing Chart of Master Reception (in Continuous Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)** | (c) |
| p.456 | Modification of **Figure 12-39. Flowchart of Master Reception (in Continuous Reception Mode)** | (c) |
| p.457 | Modification of **12.5.3 Master transmission/reception** | (c) |

**Remark** "Classification" in the above table classifies revisions as follows.
(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note, (d): Addition/change of package, part number, or management division, (e): Addition/change of related documents

(6/11)

| Page | Description | Classification |
|---|---|---|
| p.460 | Modification of **Figure 12-41. Initial Setting Procedure for Master Transmission/Reception** | (c) |
| p.463 | Modification of **Figure 12-44. Timing Chart of Master Transmission/Reception (in Single-Transmission/Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)** | (c) |
| p.464 | Modification of **Figure 12-45. Flowchart of Master Transmission/Reception (in Single-Transmission/Reception Mode)** | (c) |
| p.465 | Modification of **Figure 12-46. Timing Chart of Master Transmission/Reception (in Continuous Transmission/Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)** | (c) |
| p.466 | Modification of **Figure 12-47. Flowchart of Master Transmission/Reception (in Continuous Transmission/Reception Mode)** | (c) |
| p.467 | Modification of **12.5.4 Slave transmission** | (c) |
| p.471 | Modification of **Figure 12-50. Procedure for Stopping Slave Transmission** | (c) |
| p.472 | Modification of **Figure 12-51. Procedure for Resuming Slave Transmission** | (c) |
| p.473 | Modification of **Figure 12-52. Timing Chart of Slave Transmission (in Single-Transmission Mode) (Type 1: DAPmn = 0, CKPmn = 0)** | (c) |
| p.474 | Modification of **Figure 12-53. Flowchart of Slave Transmission (in Single-Transmission Mode)** | (c) |
| p.475 | Modification of **Figure 12-54. Timing Chart of Slave Transmission (in Continuous Transmission Mode) (Type 1: DAPmn = 0, CKPmn = 0)** | (c) |
| p.476 | Modification of Figure **12-55. Flowchart of Slave Transmission (in Continuous Transmission Mode)** | (c) |
| p.477 | Modification of **12.5.5 Slave reception** | (c) |
| p.481 | Modification of **Figure 12-58. Procedure for Stopping Slave Reception** | (c) |
| p.483 | Modification of **Figure 12-60. Timing Chart of Slave Reception (in Single-Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)** | (c) |
| p.484 | Modification of **Figure 12-61. Flowchart of Slave Reception (in Single-Reception Mode)** | (c) |
| p.485 | Modification of **12.5.6 Slave transmission/reception** | (c) |
| p.489 | Modification of **Figure 12-64. Procedure for Stopping Slave Transmission/Reception** | (c) |
| p.491 to 494 | Modification of **Figures 12-66** to **12-69** | (c) |
| p.495 to 498 | Modification of **Figures 12-70** to **12-73** | (c) |
| p.502 | Modification of **12.6 Operation of UART (UART0 to UART2) Communication** | (c) |
| p.505 | Modification of **12.6.1 UART transmission** | (c) |
| p.506 | Modification of **Figure 12-75. Example of Contents of Registers for UART Transmission of UART (UART0 to UART2) (1/2)** | (c) |
| p.508 | Modification of **Figure 12-76. Initial Setting Procedure for UART Transmission** | (c) |
| p.510 | Modification of **Figure 12-78. Procedure for Resuming UART Transmission** | (c) |
| p.511 | Modification of **Figure 12-79. Timing Chart of UART Transmission (in Single-Transmission Mode)** | (c) |
| p.512 | Modification of **Figure 12-80. Flowchart of UART Transmission (in Single-Transmission Mode)** | (c) |
| p.513 | Modification of **Figure 12-81. Timing Chart of UART Transmission (in Continuous Transmission Mode)** | (c) |

**Remark** "Classification" in the above table classifies revisions as follows.
(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note, (d): Addition/change of package, part number, or management division, (e): Addition/change of related documents

(7/11)

| Page | Description | Classification |
|---|---|---|
| p.514 | Modification of **Figure 12-82. Flowchart of UART Transmission (in Continuous Transmission Mode)** | (c) |
| p.515 | Modification of **12.6.2 UART reception** | (c) |
| p.516 | Modification of **Note** in **Figure 12-83. Example of Contents of Registers for UART Reception of UART (UART0 to UART2) (1/2)** | (c) |
| p.518 | Modification of **Figure 12-85. Procedure for Stopping UART Reception** | (c) |
| p.520 | Modification of **Figure 12-87. Timing Chart of UART Reception** | (c) |
| p.521 | Modification of **Figure 12-88. Flowchart of UART Reception** | (c) |
| p.522 | Modification of **12.6.3 SNOOZE mode function** | (c) |
| p.524 to 527 | Modification of **Figures 12-89** to **21-92** | (c) |
| p.528, 529 | Modification of figure and **Caution** in **Figure 12-93. Flowchart of SNOOZE Mode Operation (EOCm1 = 1, SSECm = 1)** | (c) |
| p.536 | Modification of **Figure 12-97. Master Transmission Operation of LIN** | (c) |
| p.545 | Modification of • 32-pin products | (c) |
| p.547 | Modification of **Note** in **12.8.1 Address field transmission** | (c) |
| p.548 | Modification of **Note** in **Figure 12-103. Example of Contents of Registers for Address Field Transmission of Simplified $I^2C$ (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21)** | (c) |
| p.553 | Modification of **12.8.2 Data transmissi**on | (c) |
| p.554 | Modification of **Figure 12-107. Example of Contents of Registers for Data Transmission of Simplified $I^2C$ (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21)** | (c) |
| p.557 | Modification of **12.8.3 Data receptio** | (c) |
| p.558 | Modification of **Figure 12-110. Example of Contents of Registers for Data Reception of Simplified $I^2C$ (IIC00, IIC01, IIC10, IIC11, IIC20, IIC21)** | (c) |
| p.565 | Modification of **Figure 12-116. Processing Procedure in Case of ACK Error in Simplified $I^2C$ Mode** | (c) |
| **CHAPTER 13 SERIAL INTERFACE IICA** | | |
| p.576, 577 | Modification of **Note** in **Figure 13-6. Format of IICA Control Register 00 (IICCTL00)** | (c) |
| p.584 | Modification of **Caution** in **Figure 13-9. Format of IICA Control Register 01 (IICCTL01) (2/2)** | (c) |
| p.585 | Modification of **Remark** in **13.3.7 IICA high-level width setting register 0 (IICWH0)** | (c) |
| p.589 | Modification of **Remark** | |
| p.601 | Modification of **13.5.13 Wakeup function** | (c) |
| p.603 | Modification of **Figure 13-24. When Operating as Master Device After Releasing STOP Mode Other than by INTIICA0** | (c) |
| p.608 | Modification of **13.5.15 Cautions** | (c) |
| p.610 | Modification of **Figure 13-28. Master Operation in Single-Master System** | (c) |
| p.611 to 613 | Modification of figure and **Note** in **Figure 13-29. Master Operation in Multi-Master System** | (c) |
| p.615 | Modification of **Figure 13-30. Slave Operation Flowchart (1)** | (c) |
| **CHAPTER 14 MULTIPLIER AND DIVIDER/MULTIPLY-ACCUMULATOR** | | |
| p.653 | Modification of figure and **Remark** in **Figure 14-1. Block Diagram of Multiplier and Divider/Multiply-Accumulator** | (c) |
| p.658 | Modification of **14.3.1 Multiplication/division control register (MDUC)** | (c) |

**Remark** "Classification" in the above table classifies revisions as follows.
(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note, (d): Addition/change of package, part number, or management division, (e): Addition/change of related documents

(8/11)

| Page | Description | Classification |
|---|---|---|
| p.658 | Modification of **Figure 14-5. Format of Multiplication/Division Control Register (MDUC)** | (c) |
| p.665 | Modification of **Figure 14-9. Timing Diagram of Multiply-Accumulation (signed) Operation** | (c) |
| **CHAPTER 15 DMA CONTROLLER** | | |
| p.668 | Modification of **15.1 Functions of DMA Controller** | (c) |
| p.681 | Modification of **Figure 15-9. Example of Setting for UART Consecutive Reception + ACK Transmission** | (c) |
| p.686 | Modification of **(4) DMA pending instruction** | (c) |
| p.687 | Modification of **(6) Operation if instructions for accessing the data flash area** | (c) |
| **CHAPTER 16 INTERRUPT FUNCTIONS** | | |
| p.691 | Modification of **Table 16-1. Interrupt Source List (3/3)** | (c) |
| p.696 | Modification of **Note 3** in **Table 16-2. Flags Corresponding to Interrupt Request Sources (3/4)** | (c) |
| p.698 | Modification of **16.3.1 Interrupt request flag registers (IF0L, IF0H, IF1L, IF1H, IF2L, IF2H)** | (c) |
| p.699 | Modification of figure and **Caution** in **Figure 16-2. Format of Interrupt Request Flag Registers (IF0L, IF0H, IF1L, IF1H, IF2L, IF2H) (2/2)** | (c) |
| p.699 | Modification of **16.3.2 Interrupt mask flag registers (MK0L, MK0H, MK1L, MK1H, MK2L, MK2H)** | (c) |
| p.700 | Modification of **Figure 16-3. Format of Interrupt Mask Flag Registers (MK0L, MK0H, MK1L, MK1H, MK2L, MK2H)** | (c) |
| p.701, 702 | Modification of figure and **Caution** in **Figure 16-4. Format of Priority Specification Flag Registers (PR00L, PR00H, PR01L, PR01H, PR02L, PR02H, PR10L, PR10H, PR11L, PR11H, PR12L, PR12H)** | (c) |
| p.703 | Modification of **Figure 16-5. Format of External Interrupt Rising Edge Enable Registers (EGP0, EGP1) and External Interrupt Falling Edge Enable Registers (EGN0, EGN1)** | (c) |
| p.705 | Modification of **16.3.5 Program status word (PSW)** | (c) |
| p.708 | Modification of **Figures 16-8** and **16-9** | (c) |
| p.710 | Modification of **Table 16-5. Relationship Between Interrupt Requests Enabled for Multiple Interrupt Servicing During Interrupt Servicing** | (c) |
| p.713 | Modification of **16.4.4 Interrupt request hold** | (c) |
| **CHAPTER 17 KEY NTERRUPT FUNCTION** | | |
| p.720 | Modification of **17.3.4 Port mode registers 0 to 2, 7, 12, 15 (PM0 to PM2, PM7, PM12, PM15)** | (c) |
| **CHAPTER 8 STANDBY FUNCTION** | | |
| p.727 | Modification of **18.2 Registers Controlling Standby Function** | (c) |
| p.727 | Modification of **18.3.1 HALT mode** | (c) |
| p.731, 732 | Modification of figure and **Note** in **Figure 18-2. HALT Mode Release by Reset** | (c) |
| p.732 | Modification of **18.3.2 STOP mode** | (c) |
| p.734, 735 | Modification of figure, **Note**, and **Caution** in **Figure 18-3. STOP Mode Release by Interrupt Request Generation** | (c) |
| p.736 | Modification of figure and **Note** in **Figure 18-4. STOP Mode Release by Reset** | (c) |
| p.738 | Modification of **Table 18-3. Operating Statuses in SNOOZE Mode** | (c) |

**Remark** "Classification" in the above table classifies revisions as follows.

(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note, (d): Addition/change of package, part number, or management division, (e): Addition/change of related documents

(9/11)

| Page | Description | Classification |
|---|---|---|
| p.739 | Modification of **Figure 18-5. When the Interrupt Request Signal is Generated in the SNOOZE Mod** and **Figure 18-6. When the Interrupt Request Signal is not Generated in the SNOOZE Mode** | (c) |
| **CHAPTER 19 RESET FUNCTION** | | |
| p.740 | Modification of **CHAPTER 19 RESET FUNCTION** | (c) |
| p.742, 743 | Modification of figure, **Note**, and **Caution** in **Figure 19-2. Timing of Reset by RESET Input** and **Figure 19-3. Timing of Reset Due to Execution of Illegal Instruction, Watchdog Timer Overflow, RAM Parity Error, or Illegal Memory Access Overflow** | (c) |
| p.744 | Modification of description and **Note** in **Table 19-1. States of Operation During Reset Period** | (c) |
| p.746 | Modification of **Figure 19-4. Format of Reset Control Flag Register (RESF)** | (c) |
| p.747 | Modification of **Table 19-3. RESF Register Status When Reset Request Is Generated** | (c) |
| p.748 | Modification of **Figure 19-5. Procedure for Checking Reset Source** | (c) |
| **CHAPTER 20 POWER-ON-RESET CIRCUIT** | | |
| p.749 | Modification of **20.1 Functions of Power-on-reset Circuit** | (c) |
| p.751 to 753 | Modification of **Figure 20-2. Timing of Generation of Internal Reset Signal by Power-on-reset Circuit and Voltage Detector** | (c) |
| **CHAPTER 21 VOLTAGE DETECTOR** | | |
| p.754 | Modification of **21.1 Functions of Voltage Detector** | (c) |
| p.755 | Modification of **Figure 21-1. Block Diagram of Voltage Detector** | (c) |
| p.756 | Modification of **Figure 21-2. Format of Voltage Detection Register (LVIM)** | (c) |
| p.757 | Modification of **Note** and **Caution** in **Figure 21-3. Format of Voltage Detection Level Select Register (LVIS)** | (c) |
| p.758, 759 | Modification of **Table 21-1. Format of User Option Byte (000C1H/010C1H)** | (c) |
| p.760 | Modification of **21.4.1 When used as reset mode** | (c) |
| p.761 | Modification of **Figure 21-4. Timing of Voltage Detector Internal Reset Signal Generation (Option Byte LVIMDS1, LVIMDS0 = 1, 1)** | (c) |
| p.762 | Modification of **21.4.2 When used as interrupt mode** | (c) |
| p.763 | Modification of **Figure 21-5. Timing of Voltage Detector Internal Interrupt Signal Generation (Option Byte LVIMDS1, LVIMDS0 = 0, 1)** | (c) |
| p.764 | Modification of **21.4.3 When used as interrupt & reset mode** | (c) |
| p.765, 767 | Modification of **Figure 21-6. Timing of Voltage Detector Reset Signal and Interrupt Signal Generation (Option Byte LVIMDS1, LVIMDS0 = 1, 0)** | (c) |
| p.768 | Modification of **Figure 21-7. Processing Procedure After an Interrupt Is Generated** | (c) |
| p.769 | Modification of **Figure 21-8. Initial Setting of Interrupt and Reset Mode** | (c) |
| p.770, 771 | Modification of **(1)**, **(2)**, and **(3)** in **21.5 Cautions for Voltage Detector** | (c) |
| **CHAPTER 22 SAFETY FUNCTIONS** | | |
| p.772 | Modification of **(6) Frequency detection function** | (c) |
| p.775 | Modification of **Figure 22-3. Flowchart of Flash Memory CRC Operation Function (High-speed CRC)** | (c) |
| p.779, 780 | Modification of **Figures 22-8** and **22-9** | (c) |
| p.782 | Modification of **Figure 22-11. Invalid Access Detection Area** | (c) |

**Remark** "Classification" in the above table classifies revisions as follows.
(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note, (d): Addition/change of package, part number, or management division, (e): Addition/change of related documents

(10/11)

| Page | Description | Classification |
|---|---|---|
| p.784 | Modification of **22.3.7 Frequency detection function** | (c) |
| p.785 | Modification of **22.3.7.1 Timer input select register 0 (TIS0)** | (c) |
| p.786 | Modification of **22.3.8 A/D test function** | (c) |
| p.787 | Modification of **Figure 22-15. Configuration of A/D Test Function** | (c) |
| p.788 | Modification of **22.3.8.1 A/D test register (ADTES)** | (c) |
| **CHAPTER 24 OPTION BYTE** | | |
| p.792, 793 | Modification of **24.1.1 User option byte (000C0H to 000C2H/010C0H to 010C2H)** | (c) |
| p.795, 796 | Modification of **Figure 24-2. Format of User Option Byte (000C1H/010C1H)** | (c) |
| p.797 | Modification of **Figure 24-3. Format of Option Byte (000C2H/010C2H)** | (c) |
| p.799 | Modification of **24.4 Setting of Option Byte** | (c) |
| **CHAPTER 25 FLASH MEMORY** | | |
| P.800 | Modification of **CHAPTER 25 FLASH MEMORY** | (c) |
| p.803 | Modification of **Table 25-1. Wiring between RL78/G1A and Dedicated Flash Memory Programmer** | (c) |
| p.804 to 806 | Modification of **Figures 25-1** to **25-4** | (c) |
| p.803, 805, 806 | Modification of **Tables 25-1** to **25-3** | (c) |
| p.807 | Modification of **25.3 Connection of Pins on Board** and 25.3.1 P40/TOOL0 pin | (c) |
| p.808 | Modification of **25.3.4 REGC pin** | (c) |
| p.810 | Modification of **25.4.2 Flash memory programming mode** | (c) |
| p.811 | Modification of **Table 25-5. Programming Modes and Voltages at Which Data Can Be Written, Erased, or Verified** | (c) |
| p.812 | Modification of **25.4.4 Communication commands** and **Table 25-7. Flash Memory Control Commands** | (c) |
| p.814 | Modification of **25.5 Processing Time for Each Command When PG-FP5 Is in Use (Reference Value)** | (c) |
| p.815 | Modification of **25.6 Self-Programming** | (c) |
| p.820 | Modification of **25.7 Security Settings** | (c) |
| p.821 | Modification of **Caution** in **Table 25-13. Setting Security in Each Programming Mode** | (c) |
| p.822 | Modification of **25.8.1 Data flash overview** | (c) |
| p.823 | Modification of **25.8.3 Procedure for accessing data flash memory** | (c) |
| **CHAPTER 26 ON-CHIP DEBUG FUNCTION** | | |
| p.824 | Modification of figure and **Remark** in **Figure 26-1. Connection Example of E1 On-chip Debugging Emulator** | (c) |
| **CHAPTER 28 INSTRUCTION SET** | | |
| p.830 | Modification of **CHAPTER 28 INSTRUCTION SET** | (c) |
| p.833 to 849 | Modification of **Note** in **Table 28-5. Operation List** | (c) |
| **CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C)** | | |
| p.853 | Modification of **29.2.1 X1, XT1 oscillator characteristics** | (c) |
| p.864 | Modification of **Note 1** in **29.3.2 Supply current characteristics** | (c) |
| p.866, 867 | Modification of **Minimum Instruction Execution Time during Main System Clock Operation** | (c) |

**Remark** "Classification" in the above table classifies revisions as follows.
(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note, (d): Addition/change of package, part number, or management division, (e): Addition/change of related documents

(11/11)

| Page | Description | Classification |
|---|---|---|
| p.868 | Modification of **AC Timing Test Points** and **TI/TO Timing** | (c) |
| p.870 | Modification of **AC Timing Test Points** in **29.5 Peripheral Functions Characteristics** | (c) |
| p. 879, 880, 882 to 884, 887, 891 | Modification of **Caution** | (c) |
| p.896 to 901 | Modification of **29.6.1 A/D converter characteristics** | (c) |
| p.904 | Modification of **29.7 Data Memory STOP Mode Low Supply Voltage Data Retention Characteristics** | (c) |
| p.904 | Modification of **29.8 Flash Memory Programming Characteristics** | (c) |
| p.905 | Modification of **29.10 Timing Specs for Switching Flash Memory Programming Modes** | (c) |
| **CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to +105°C)** | | |
| p.909 | Modification of **30.2.1 X1, XT1 oscillator characteristics** | (c) |
| p.910, 911 | Modification of **30.3.1 Pin characteristics** | (c) |
| p.915 | Modification of **30.3.2 Supply current characteristics** | (c) |
| p.922 | Modification of **Minimum Instruction Execution Time during Main System Clock Operation** | (c) |
| p.923 | Modification of **AC Timing Test Points** | (c) |
| p.925 | Modification of **AC Timing Test Points** | (c) |
| p.925 | Modification of **30.5.1 Serial array unit** | |
| p.931, 932 | Modification of **Caution** in **30.5.1 (5)** | (c) |
| p.934, 935 | Modification of **Caution** in **30.5.1 (6)** | (c) |
| p.937 | Modification of **Caution** in **30.5.1 (7)** | (c) |
| p.941 | Modification of **Caution** in **30.5.1 (8)** | (c) |
| p.944 to 948 | Modification of **30.6.1 (1)** to **(5)** | (c) |
| p.950 | Modification of **30.7 Data Memory STOP Mode Low Supply Voltage Data Retention Characteristics** | (c) |
| p.951 | Modification of **30.10 Timing Specs for Switching Flash Memory Programming Modes** | (c) |
| **CHAPTER 31 PACKAGE DRAWINGS** | | |
| p.952 to 954, 956 | Modification of **Products** | (c) |

**Remark** “Classification” in the above table classifies revisions as follows.
(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note, (d): Addition/change of package, part number, or management division, (e): Addition/change of related documents

## A.2 Revision History of Preceding Editions

Here is the revision history of the preceding editions. Chapter indicates the chapter of each edition.

(1/5)

| Page | Description | Classification |
|---|---|---|
| R01UH0305EJ0100 → R01UH0305EJ0110 | | |
| **CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = −40 to +105°C) (TARGET)** | | |
| p.952 | Modification of $I_{SNOZ}$ and **note 7** in **30.3.2 Supply current characteristics** | (c) |
| R01UH0305EJ0003 → R01UH0305EJ0100 | | |
| **CHAPTER 1 OUTLINE** | | |
| p.1 | Modification of **1.1 Features** | (c) |
| p.3 | Modification of figure and addition of **caution** to **Figure 1-1. Part Number, Memory Size, and Package of RL78/G1A** | (d) |
| p.4 | Modification of table and addition of **caution** in **1.2 List of Part Numbers** | (d) |
| p.14 | Modification of **1.5.3 48-pin products** | (a) |
| p.16 | Modification of description and **note 2** in **1.6 Outline of Functions** | (c) |
| **CHAPTER 3 CPU ARCHITECTURE** | | |
| p.36 | Modification of description in **3.1 Memory Space** | (c) |
| p.37 to 40 | Modification of f **note 1** to **Figures 3-1 to 3-4** | (c) |
| p.39, 40 | Modification of **note 2** in **Figures 3-3, 3-4** | (c) |
| p.47 | Deletion of **caution 2** in **Figure 3-5. Format of Processor Mode Control Register (PMC)** | (c) |
| p.48 | Addition of **caution 2** and modification of **caution 3** in **3.1.3 Internal data memory space** | (c) |
| p.50 to 53 | Modification of **note 1** in **Figures 3-6 to 3-9** | (c) |
| p.52, 53 | Modification of **note 2** to **Figures 3-8, 3-9** | (c) |
| p.54 | Modification of description in **3.2.1 (1) Program counter (PC)** and **3.2.1 (2) (b) Zero flag (Z)** | (c) |
| p.55 | Modification of description in **3.2.1 (3) Stack pointer (SP)** | (c) |
| p.56 | Modification of **3.2.2 General-purpose registers** and **Figure 3-13. Configuration of General-Purpose Registers** | (c) |
| p.57 | Addition of **Figure 3-15 Extension of Data Area Which Can Be Accessed** | (c) |
| p.58 | Modification of description in **3.2.4 Special function registers (SFRs)** | (c) |
| p.64 | Modification of description in **3.2.5 Extended special function registers (2nd SFRs)** | (c) |
| p.66 | Modification of description and **note** in **Table 3-6. Extended SFR (2nd SFR) List** | (c) |
| p.71, 73 to 87 | Modification of Figure 3-16 to 3-18, 3-20 to 3-43 | (c) |
| p.74 | Modification of [Operand format] in **3.4.1 Implied addressing** | (c) |
| p.75 | Modification of [Operand format] in **3.4.3 Direct addressing** | (c) |
| p.77 | Modification of [Operand format] in **3.4.5 SFR addressing** | (c) |
| p.79 | Modification of [Function] in **3.4.7 Based addressing** | (c) |
| p.84 | Modification from [Operand format] to [Description format], modification of [Function] and [Description format] in **3.4.9 Stack addressing** | (c) |

**Remark** "Classification" in the above table classifies revisions as follows.
(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note, (d): Addition/change of package, part number, or management division, (e): Addition/change of related documents

(2/5)

| Page | Description | Classification |
|---|---|---|
| **CHAPTER 5 CLOCK GENERATOR** | | |
| p.162 | Addition of description to **5.1 (1) <2> High-speed on-chip oscillator** | (c) |
| p.165 | Modification of **Figure 5-1. Block Diagram of Clock Generator** | (c) |
| p.167 | Modification of **caution 3** in **Figure 5-2. Format of Clock Operation Mode Control Register (CMC)** | (c) |
| p.185 | Modification of description in **5.4.3 High-speed on-chip oscillator** | (c) |
| p.187 | Modification of **Figure 5-14. Clock Generator Operation When Power Supply Voltage Is Turned On** | (c) |
| p.188 | Modification of description in **5.6.1 Example of setting high-speed on-chip oscillator** | (c) |
| p.189 | Modification of description in **5.6.2 Example of setting X1 oscillation clock** | (c) |
| p.191 | Modification of **Figure 5-15. CPU Clock Status Transition Diagram** | (c) |
| p.194 | Modification of description in **Table 5-3. CPU Clock Transition and SFR Register Setting Examples** | (c) |
| p.202 to 204 | Addition of **note** and modification of **remark** in **5.7 (1) X1 oscillation:** | (c) |
| **CHAPTER 6 TIMER ARRAY UNIT** | | |
| p.214 to 216 | Modification of **Figures 6-2** to **6-6** | (c) |
| p.239 | Modification of **note** in **6.3.14 Noise filter enable register 1 (NFEN1)** | (c) |
| p.251 | Modification of **Figure 6-29. Operation Timing (In Capture Mode : Input Pulse Interval Measurement)** | (c) |
| p.263, 264 | Addition of **6.7 Timer Input (TImn) Cntorol** | (c) |
| p.296 | Modification of description and addition of **note** to **Figure 6-69. Example of Set Contents of Registers When One-Shot Pulse Output Function Is Used (Master Channel)** | (c) |
| p.297 | Modification of description and **note** in **Figure 6-70. Example of Set Contents of Registers When One-Shot Pulse Output Function Is Used (Slave Channel)** | (c) |
| p.299 | Modification of **Figure 6-71. Operation Procedure of One-Shot Pulse Output Function** | (c) |
| p.303 | Addition of **note** to **Figure 6-74. Example of Set Contents of Registers When PWM Function (Master Channel) Is Used** | (c) |
| p.304 | Modification of description and **note** in **Figure 6-75. Example of Set Contents of Registers When PWM Function (Slave Channel) Is Used** | (c) |
| p.311 | Addition **note** to **Figure 6-79. Example of Set Contents of Registers When Multiple PWM Output Function (Master Channel) Is Used** | (c) |
| p.312 | Modification of description and **note** in **Figure 6-80. Example of Set Contents of Registers When Multiple PWM Output Function (Slave Channel) Is Used (output two types of PWMs)** | (c) |
| **CHAPTER 7 REAL-TIME CLOCK** | | |
| p.336 | Modification of **Figure 7-23. Operation when (DEV, F6, F5, F4, F3, F2, F1, F0) = (1, 1, 1, 0, 1, 1, 1, 0)** | (c) |
| **CHAPTER 8 12-BIT INTERVAL TIMER** | | |
| p.343 | Modification of **Figure 8-2. Format of Peripheral Enable Register 0 (PER0)** | (c) |
| p.344 | Modification of description in **8.3.2 Operation speed mode control register (OSMC)** | (c) |
| p.346 | Modification of **Figure 8-5. 12-bit Interval Timer Operation Timing (ITCMP11 to ITCMP0 = 0FFH, count clock: $f_{SUB}$ = 32.768 kHz)** | (c) |

**Remark** "Classification" in the above table classifies revisions as follows.
(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note, (d): Addition/change of package, part number, or management division, (e): Addition/change of related documents

(3/5)

| Page | Description | Classification |
|---|---|---|
| **CHAPTER 11 A/D CONVERTER** | | |
| p.361 | Modification of **Figure 11-1. Block Diagram of A/D Converter** | (c) |
| p.365 | Modification of **caution 1** in **Figure 11-2. Format of Peripheral Enable Register 0 (PER0)** | (c) |
| p.367 | Modification of **Table 11-1. Settings of ADCS and ADCE Bits** | (c) |
| p.378 | Modification of **caution 1** in **Figure 11-6. Format of A/D Converter Mode Register 1 (ADM1)** | (c) |
| p.379, 380 | Modification of **caution** in **Figure 11-7. Format of A/D Converter Mode Register 2 (ADM2)** | (c) |
| p.385 | Modification of **caution 5** in **Figure 11-11. Format of Analog Input Channel Specification Register (ADS)** | (c) |
| **CHAPTER 12 SERIAL ARRAY UNIT** | | |
| p.424 | Modification of **note 1** in **12.1.1 3-wire serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21)** | (c) |
| p.449 | Modification of **12.3.14 Serial standby control register 0 (SSC0)**, and addition of **Table 12-2. Interrupt Using the UART Reception with SNOOZE Mode** | (c) |
| p.451 | Modification of description in **12.3.16 Noise filter enable register 0 (NFEN0)** | (c) |
| p.520 | Modification of figure and addition of **note 2** to **Figure 12-72. Timing Chart of SNOOZE Mode Operation (once startup) (Type 1: DAPmn = 0, CKPmn = 0)** | (c) |
| p.522 | Modification of figure and addition of **note 2** to **Figure 12-74. Timing Chart of SNOOZE Mode Operation (continuous startup) (Type 1: DAPmn = 0, CKPmn = 0)** | (c) |
| p.547 | Modification of description and **caution 2** and addition of **cautions 3, 4** to **12.6.3 SNOOZE mode function** | (c) |
| p.548 | Addition of **Table 12-3. UART Reception Baud Rate Setting in the SNOOZE mode** | (c) |
| p.549 | Addition of description in **12.6.3 (1) SNOOZE mode operation (EOCm1 = 0, SSECm = 0/1)** | (c) |
| p.550 | Addition of description in **12.6.3 (2) SNOOZE mode operation (EOCm1 = 1, SSECm = 0: error interrupt (INTSRE0) generation enable)** | (c) |
| p.551 | Modification of **Figure 12-93. Flowchart of SNOOZE Mode Operation (EOCm1 = 0, SSECm = 0/1 or EOCm1 = 1, SSECm = 0)** | (c) |
| p.552 | Modification of **12.6.3 (3) SNOOZE mode operation (EOCm1 = 1, SSECm = 1: error interrupt (INTSRE0) generation stop)** | (c) |
| p.553, 554 | Modification of **Figure 12-95. Flowchart of SNOOZE Mode Operation (EOCm1 = 1, SSECm = 1)** | (c) |
| p.557 | Modification of **12.6.4 (2) Baud rate error during transmission** | (c) |
| **CHAPTER 14 MULTIPLIER AND DIVIDER/MULTIPLY-ACCUMULATOR** | | |
| p.685 | Modification of description in **14.4.1 Multiplication (unsigned) operation** | (c) |
| p.686 | Modification of description in **14.4.2 Multiplication (signed) operation** | (c) |
| p.687 | Modification of description in **14.4.3 Multiply-accumulation (unsigned) operation** | (c) |
| p.689 | Modification of description in **14.4.4 Multiply-accumulation (signed) operation** | (c) |
| p.690 | Modification of **Figure 14-9. Timing Diagram of Multiply-Accumulation (signed) Operation (2 × 3 + (–4) = 2 → 32767 × (–1) + (–2147483647) = –2147450882 (overflow occurs.))** | (c) |
| p.691 | Modification of description in **14.4.5 Division operation** | (c) |
| **CHAPTER 16 INTERRUPT FUNCTION** | | |
| p.715 | Addition of **Note 5** to **Table 16-1. Interrupt Source List** | (a) |

**Remark** "Classification" in the above table classifies revisions as follows.

(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note, (d): Addition/change of package, part number, or management division, (e): Addition/change of related documents

(4/5)

| Page | Description | Classification |
|---|---|---|
| **CHAPTER 18 STANDBY FUNCTION** | | |
| p.758 | Modification of **Table 18-1. Operating Statuses in HALT Mode** | (c) |
| p.760, 761 | Modification of figure and **note** in **Figure 18-4. HALT Mode Release by Reset** | (c) |
| p.762 | Modification of **caution 1** in **18.3.2 (1) STOP mode setting and operating statuses** | (c) |
| p.764 | Deletion of **caution 1** in **Table 18-2. Operating Statuses in STOP Mode** | (c) |
| p.764, 765 | Modification of **note 2** and addition of **remark 2** to **Figure 18-5. STOP Mode Release by Interrupt Request Generation** | (c) |
| p.766 | Modification of figure and **note** in **Figure 18-6. STOP Mode Release by Reset** | (c) |
| p.767 | Modification of description in **18.3.3 (1) SNOOZE mode setting and operating statuses** | (c) |
| **CHAPTER 19 RESET FUNCTION** | | |
| p.769 | Modification of description and **cautions 1, 3** in **CHAPTER 19 RESET FUNCTION** | (c) |
| p.771, 772 | Modification of figure and **note 2** in **Figures 19-2**, **19-3** | (c) |
| p.773 | Modification of description in **Table 19-1. Operation Statuses During Reset Period** | (c) |
| p.774 | Modification of description and **note 2** in **Table 19-2. Hardware Statuses After Reset Acknowledgment** | (c) |
| **CHAPTER 20 POWER-ON-RESET CIRCUIT** | | |
| p.781 | Modification of description and addition of **note** to **20.1 Functions of Power-on-reset Circuit** | (c) |
| p.783 to 785 | Modification of figure and **notes** in **Figure 20-2. Timing of Generation of Internal Reset Signal by Power-on-reset Circuit and Voltage Detector** | (c) |
| **CHAPTER 21 VOLTAGE DETECTOR** | | |
| p.788 | Modification of description in **21.1 Functions of Voltage Detector** | (c) |
| p.795 | Modification of **Figure 21-4. Timing of Voltage Detector Internal Reset Signal Generation (Option Byte LVIMDS1, LVIMDS0 = 1, 1)** | (c) |
| p.796 | Addition of **note** to **21.4.2 When used as interrupt mode** | (c) |
| p.797 | Modification of figure and addition of **notes 2, 3** to **Figure 21-5. Timing of Voltage Detector Internal Interrupt Signal Generation (Option Byte LVIMDS1, LVIMDS0 = 0, 1)** | (c) |
| p.798 | Addition of **note** to **21.4.3 When used as interrupt and reset mode** | (c) |
| p.799 to 802 | Modification of figure and addition of **note 4** to **Figure 21-6. Timing of Voltage Detector Reset Signal and Interrupt Signal Generation (Option Byte LVIMDS1, LVIMDS0 = 1, 0)** | (c) |
| p.802 | Modification of **Figure 21-7. Processing Procedure After an Interrupt Is Generated** | (c) |
| p.803 | Modification of **Figure 21-8. Initial Setting of Interrupt and Reset Mode** | (c) |
| **CHAPTER 22 SAFETY FUNCTIONS** | | |
| p.806 | Modification of **remark** in **22.1 Overview of Safety Functions** | (c) |
| p.812 | Modification of **caution** and **remarks 1 to 4** in **Figure 22-7. Format of RAM Parity Error Control Register (RPECTL)** | (c) |
| p.815 | Addition of figure and **note** to **Figure 22-10. Invalid access detection area** | (c) |
| **CHAPTER 24 OPTION BYTE** | | |
| p.824 | Modification of description and addition of **caution** to **24.1 Functions of Option Bytes** | (c) |
| p.825 | Modification of description in **24.1.1 User option byte (000C0H to 000C2H/010C0H to 010C2H)** | (c) |
| p.826 | Modification of **caution** in **Figure 24-1. Format of User Option Byte (000C0H/010C0H)** | (c) |
| p.827, 828 | Addition of **caution 2** to **Figure 24-2. Format of User Option Byte (000C1H/010C1H)** | (c) |
| p.829 | Modification of description and **caution** in **Figure 24-3. Format of User Option Byte (000C2H/010C2H)** | (c) |

**Remark** "Classification" in the above table classifies revisions as follows.

(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note, (d): Addition/change of package, part number, or management division, (e): Addition/change of related documents

(5/5)

| Page | Description | Classification |
|---|---|---|
| **CHAPTER 28 INSTRUCTION SET** | | |
| p.880 | Modification of **Table 28-5. Operation List** | (c) |
| **CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C)** | | |
| Though out | Deletion of target in ELECTRICAL SPECIFICATIONS | (b) |
| p.884 | Addition of target products | (d) |
| p.885 | Modification of description and addition of **remark 3** to **29.1 Absolute Maximum Ratings** | (c) |
| p.887 | Modification of description and **note** in **29.2.1 X1, XT1 oscillator characteristics** | (c) |
| p.888 | Modification of **caution** in **29.2.2 On-chip oscillator characteristics** | (c) |
| p.889, 890 | Modification of **note 3** in **29.3.1 Pin characteristics** | (c) |
| p.895, 897 to 899 | Modification of description and **notes** in **29.3.2 Supply current characteristics** | (c) |
| p.900, 901 | Addition of description and modification of **remark** in **29.4 AC Characteristics** | (c) |
| p.903 to 908, 910 to 913, 915 to 917, 920, 923, 924 | Modification of **29.5.1 Serial array unit** | (c) |
| p.926, 927 | Modification to **29.5.2 Serial interface IICA** | (c) |
| p.930 to 933 | Modification of **29.6.1 A/D converter characteristics** | (c) |
| p.933 | Addition of **note** to **29.6.3 POR circuit characteristics** | (c) |
| p.934, 935 | Modification of LVD Detection Voltage of Interrupt & Reset Mode and Supply Voltage Rise Slope in **29.6.4 LVD circuit characteristics** | (c) |
| p.937 | Modification of **29.7 Data Memory STOP Mode Low Supply Voltage Data Retention Characteristics** | (c) |
| p.938 | Modification of **29.9 Timing Specs for Switching Flash Memory Programming Modes** | (c) |
| **CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to +105°C) (TARGET)** | | |
| p.939 | Addition of target products | (d) |
| p.940 | Addition of **note 5** to **30.1 Absolute Maximum Ratings** | (c) |
| p.942 | Modification of value and **note** in **30.2.1 X1, XT1 oscillator characteristics** | (c) |
| p.943 | Modification of **caution** in **30.2.2 On-chip oscillator characteristics** | (c) |
| p.944, 945 | Modification of **note 3** in **30.3.1 Pin characteristics** | (c) |
| p.949, 950, 952, 953 | Modification of description and **notes** in **30.3.2 Supply current characteristics** | (c) |
| p.954 | Modification of description and addition of **remark** in **30.4 AC Characteristics** | (c) |
| p.957, 959, 963, 964, 972, 973 | Modification of **30.5.1 Serial array unit** | (c) |
| p.976 to 980 | Modification of **30.6.1 A/D converter characteristics** | (b) |
| p.980 | Addition of **note** to **30.6.3 POR circuit characteristics** | (c) |
| p.981 | Modification of LVD Detection Voltage of Interrupt & Reset Mode and Supply Voltage Rise Slope in **30.6.4 LVD circuit characteristics** | (c) |
| p.983 | Modification of **30.9 Timing Specs for Switching Flash Memory Programming Modes** | (c) |
| **CHAPTER 31 PACKAGE DRAWINGS** | | |
| p.985, 987 | Addition of the products | (d) |

**Remark** "Classification" in the above table classifies revisions as follows.

(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note, (d): Addition/change of package, part number, or management division, (e): Addition/change of related documents

(1/8)

| Edition | Description | Chapter |
|---|---|---|
| Rev.0.03 | Renamed interval timer (unit) to 12-bit interval timer | Though out |
| | Renamed $V_{LVI}$, $V_{LVIH}$, $V_{LVIL}$ to $V_{LVD}$, $V_{LVDH}$, $V_{LVDL}$ (LVD detection voltage) | |
| | Renamed interrupt source of RAM parity error (RAMTOP) to RPE | |
| | Modification of **1.2 Ordering Information** | **CHAPTER 1 OUTLINE** |
| | Addition of **Figure 1-1. Part Number, Memory Size, and Package of RL78/G1A** | |
| | Modification of **1.5.1 25-pin products** to **1.5.4 64-pin products** | |
| | Modification of **Table 2-1. Pin I/O Buffer Power Supplies** | **CHAPTER 2 PIN FUNCTIONS** |
| | Modification of **2.1 Port Function** | |
| | Modification of description in **2.2 Functions other than port pins (Deletion of Description of Port Function)** | |
| | Addition of **remark** to **2.3 Pin I/O Circuits and Recommended Connection of Unused Pins** | |
| | Change of **Table 2-3. Connection of Unused Pins (64-pin products) (2/2)** | |
| | Addition of **note 1** to **Figures 3-1, 3-2** | **CHAPTER 3 CPU ARCHITECTURE** |
| | Addition of **caution** to **Figures 3-1 to 3-4** | |
| | Modification of **note** in **Figures 3-3, 3-4** | |
| | Addition of **remark** to **Table 3-1. Correspondence Between Address Values and Block Numbers in Flash Memory** | |
| | Modification of **caution 2** in **3.1.3 Internal data memory space** | |
| | Addition of **note 1** to **Figures 3-6, 3-7** | |
| | Addition of **caution** to **Figures 3-6 to 3-9** | |
| | Addition of **note 1** to **Figures 3-8, 3-9** | |
| | Modification of **caution 3** in **3.2.1 (3) Stack pointer (SP)** | |
| | Deletion of **caution 2** in **3.2.2 General-purpose registers** | |
| | Modification of **4.1 Port Functions** | **CHAPTER 4 PORT FUNCTIONS** |
| | Modification of block diagrams | |
| | Addition of description to **4.2.3 Port 2** | |
| | Addition of description to **4.2.12 Port 15** | |
| | Addition of **caution** to **4.3 Registers Controlling Port Function** | |
| | Modification of **Figure 4-39. Format of Port Register (64-pin products)** | |
| | Modification of description and addition of caution to **4.3.3 Pull-up resistor option registers (PUxx)** | |
| | Addition of **4.3.5 Port output mode registers (POMxx)** | |
| | Addition of **cautions 1** and **2** to **Figure 4-43. Format of Port Mode Control Register** | |
| | Addition of **caution 1** to **Figure 4-44. Format of A/D Port Configuration Register (ADPC)** | |
| | Modification of description in **4.3.8 Peripheral I/O redirection register (PIOR)** | |

(2/8)

| Edition | Description | Chapter |
|---|---|---|
| Rev.0.03 | Addition of **remark** to **4.3.9 Global digital input disable register (GDIDIS)**<br>Modification of description to **4.3.10 Global analog input disable register (GAIDIS)** | **CHAPTER 4 PORT FUNCTIONS** |
| | Modification of description in **4.4.1 (2) Input mode** and **4.4.3 (2) Input mode** | |
| | Addition of description to **4.4.4 (2) Setting procedure when using I/O pins of IIC00, IIC10, and IIC20 functions** | |
| | Addition of **caution** to **4.5 Settings of Port Mode Register, and Output Latch When Using Alternate Function** | |
| | Addition of **4.6.2 Notes on specifying the pin settings** | |
| | Addition of **5.1 (1) <2> High-speed on-chip oscillator** | **CHAPTER 5 CLOCK GENERATOR** |
| | Modification of **Figure 5-1. Block Diagram of Clock Generator** | |
| | Modification of **caution 1** and addition of **cautions 4 to 6** to **Figure 5-2. Format of Clock Operation Mode Control Register (CMC)** | |
| | Modification of **cautions 1 to 3** in **Figure 5-9. Format of High-speed On-chip Oscillator Frequency Select Register (HOCODIV)** | |
| | Modification of **note 3** in **Figure 5-14. Clock Generator Operation When Power Supply Voltage Is Turned On** | |
| | Addition of description to **5.6.2 Example of setting X1 oscillation clock** | |
| | Addition of description to **Figure 5-15. CPU Clock Status Transition Diagram** | |
| | Modification of **Table 5-3. CPU Clock Transition and SFR Register Setting Examples** | |
| | Modification and deletion of description in **Table 5-4. Changing CPU Clock** | |
| | Modification of **remark 2** to **5.6.6 Time required for switchover of CPU clock and system clock** | |
| | Addition of **5.7 Recommended Oscillator Constants** | |
| | Modification of description in **6.1.1 (7) Delay counter** | **CHAPTER 6 TIMER ARRAY UNIT** |
| | Modification of **caution** in **6.1.2 (3) Multiple PWM (Pulse Width Modulation) output** | |
| | Modification of **Figure 6-2. Internal Block Diagram of Channels of Timer Array Unit 0, 2, 4, 6** | |
| | Addition of **Figures 6-3** to **6-6** | |
| | Modification of **Table 6-3. Timer Count Register mn (TCRmn) Read Value in Various Operation Modes** | |
| | Modification of **note** and **remark 2** and addition of **caution** to **Figure 6-11. Format of Timer Clock Select register m (TPSm)** | |
| | Modification of **Figure 6-12. Format of Timer Mode Register mn (TMRmn)** | |
| | Modification of description in **Figure 6-18. Format of Timer Output Enable register m (TOEm)** | |
| | Modification of description in **6.5.1 (1) When operation clock ($f_{MCK}$) specified by the CKSmn0 and CKSmn1 bits is selected (CCSmn = 0)** | |
| | Modification of description in **Table 6-6. Operations from Count Operation Enabled State to Timer count Register mn (TCRmn) Count Start** | |
| | Addition of title and **remark** to **6.5.3 Operation of counter** | |
| | Modification of description, **remark** and addition **note** to **Figure 6-29. Start Timing (In Capture Mode : Input Pulse Interval Measurement)** | |

(3/8)

| Edition | Description | Chapter |
|---|---|---|
| Rev.0.03 | Modification of **remark 1** in **Figure 6-31. Operation Timing (In Capture & One-count Mode : High-level Width Measurement)** | **CHAPTER 6 TIMER ARRAY UNIT** |
| | Modification of description in **6.6.2 TOmn Pin Output Setting** | |
| | Modification of **Figures 6-34** to **6-36** | |
| | Modification of description in **Figures 6-43, 6-47, 6-55, 6-59, 6-63, 6-68, 6-78 Example of Set Contents of Registers** | |
| | Modification of **Figures 6-45, 6-49, 6-57, 6-61, 6-65 Block Diagram** | |
| | Modification of **Figures 6-48**, **6-52, 6-56, 6-60, 6-64, 6-69, 6-79 Operation Procedure** | |
| | Modification of **remark** in **6.8.3 Operation as multiple PWM output function** | |
| | Modification of **7.4.2 Shifting to HALT/STOP mode after starting operation** | **CHAPTER 7 REAL-TIME CLOCK** |
| | Addition of **caution** in **9.1 Functions of Clock Output/Buzzer Output Controller** | **CHAPTER 9 CLOCK OUTPUT/BUZZER OUTPUT CONTROLLER** |
| | Modification of **Figure 9-2. Format of Clock Output Select Register n (CKSn)** | |
| | Addition of **9.5 Cautions of clock output/buzzer output controller** | |
| | Modification of description in **10.1 Functions of Watchdog Timer**, **10.4.4 Setting watchdog timer interval interrupt** | **CHAPTER 10 WATCHDOG TIMER** |
| | Modification of **Figure 10-1 . Block Diagram of Watchdog Timer** | |
| | **Figure 11-1. Block Diagram of A/D Converter** | **CHAPTER 11 A/D CONVERTER** |
| | Deletion of **note 3** and addition of **cautions 1** and **2** to **Figure 11-3. Format of A/D Converter Mode Register 0 (ADM0)** | |
| | Modification of description and addition of **note** and **caution** in **Table 11-3. A/D Conversion Time Selection** | |
| | Modification of description and addition of **note** to **Figure 11-7. Format of A/D Converter Mode Register 2 (ADM2)** | |
| | Addition of **note** to **11.3.5 10-bit A/D conversion result register (ADCR)**, and **11.3.6 8-bit A/D conversion result register (ADCRH)** | |
| | Addition of **note** and **caution 10** to **Figure 11-11. Format of Analog Input Channel Specification Register (ADS)** | |
| | Addition of **note** and **caution** to **11. 3.10 A/D test register (ADTES)** | |
| | Addition of **caution** to **11.3.12 Port mode control registers 0, 1, 3to 5, 7, 12 (PMC0, PMC1, PMC3 to PMC5, PMC12)** | |
| | Addition of **note 1** to **11.4 A/D Converter Conversion Operations** | |
| | Addition of **caution** to **11.7.4 Setup when temperature sensor output voltage/internal reference voltage is selected (example for software trigger mode and one-shot conversion mode)** | |
| | Modification of description in **11.8 SNOOZE Mode Function** | |
| | Modification of description in **11.9 (1) Resolution** | |
| | Addition of **caution** to **11.10 (2) Input range of ANI0 to ANI14 and ANI16 to ANI26 pins** | |
| | Modification of description in **Figure 11-46. Analog Input Pin Connection**<br>Modification of description in **11.10 (5) Analog input (ANIn) pins**<br>Modification of description in **11.10 (6) Input impedance of analog input (ANIn) pins** | |
| | Modification of value in **Table 11-6. Resistance and Capacitance Values of Equivalent Circuit (Reference Values)** | |

(4/8)

| Edition | Description | Chapter |
|---|---|---|
| Rev.0.03 | Addition of **note 1** to **12.1.1 3-wire serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21)** | **CHAPTER 12 SERIAL ARRAY UNIT** |
| | Addition of **note** to **12.1.2 UART (UART0 to UART2)** | |
| | Modification of **Figures 12-1** and **12-2 block diagram of the serial array unit 0, 1** | |
| | Modification of description in **12.2.2 Lower 8/9 bits of the serial data register mn (SDRmn)** | |
| | Modification of **caution 1** in **Figure 12-5. Format of Peripheral Enable Register 0 (PER0)** | |
| | Modification of description in **Figure 12-8. Format of Serial Communication Operation Setting Register mn (SCRmn)** | |
| | Modification of description in **12.3.5 Higher 7 bits of the serial data register mn (SDRmn)** | |
| | Addition of **note** and **caution 2** to **Figure 12-12. Format of Serial Channel Start Register m (SSm)** | |
| | Addition of **note** to **Figure 12-13. Format of Serial Channel Stop Register m (STm)** | |
| | Modification of description and addition of **caution** to **Figure 12-18. Format of Serial Standby Control Register m (SSCm)** | |
| | Modification of description in **12.3.18 Port output mode registers 0, 1, 5, 7 (POM0, POM1, POM5, POM7)** | |
| | Addition of description to **12.3.19 Port mode registers 0, 1, 3, 5, 7 (PM0, PM1, PM3, PM5, PM7)** | |
| | Modification of **note 1** in **12.5 Operation of 3-Wire Serial I/O (CSI00, CSI01, CSI10, CSI11, CSI20, CSI21) Communication** | |
| | Modification of description in **Figure 12-77, 85, 105, 109, 112 (Example of Contents of Registers)** | |
| | Modification of **Figure 12-28, 29, 36, 37, 44, 45, 52, 53, 60, 61, 66, 67, 79, 80, 82, 84, 87, 88, 90, 93, 95, 106, 108, 111, 114, 116 (flow chart)** | |
| | Modification of description in **12.5.7 SNOOZE mode function** | |
| | Modification of **caution** in **Figures 12-72 and 12-74** | |
| | Modification of description in **Table 12-2. Selection of Operation Clock For 3-Wire Serial I/O** | |
| | Addition of description to **12.6 Operation of UART (UART0 to UART3) Communication** | |
| | Modification of description in **12.6.1 UART transmission** and **12.6.2 UART reception** | |
| | Modification of **caution** in **Figure 12-86. Initial Setting Procedure for UART Reception** | |
| | Addition of description and modification of **caution** to **12.6.3 SNOOZE mode function**<br>Modification of **note** and **caution** in **Figure 12-91. Timing Chart of SNOOZE Mode Operation (Normal operation mode)** | |
| | Modification of **caution** in **Figure 12-92. Timing Chart of SNOOZE Mode Operation (Abnormal Operation <1>)** | |

(5/8)

| Edition | Description | Chapter |
|---|---|---|
| Rev.0.03 | Modification of **note 1** and **caution 1** in **Figure 12-94. Timing Chart of SNOOZE Mode Operation (Abnormal Operation <2>)** | **CHAPTER 12 SERIAL ARRAY UNIT** |
| | Modification of description in **Table 12-3. Selection of Operation Clock For UART** | |
| | Modification of description and **note** in **12.7.1 LIN transmission** and **12.7.2 LIN reception** | |
| | Modification of **Figure 12-99. Master Transmission Operation of LIN** | |
| | Modification of **Figure 12-100. Flowchart for LIN Transmission** | |
| | Modification of **Figure 12-101. Reception Operation of LIN** | |
| | Modification of **Figure 12-102. Flowchart for LIN Reception** | |
| | Modification of description in **12.8.1 Address field transmission**, **12.8.2 Data transmission**, and **12.8.3 Data reception** | |
| | Addition of **caution** of description to **12.8.5 Calculating transfer rate** | |
| | Modification of description in **example of setting an $I^2C$ transfer rate** | |
| | Modification of **13.5.17 (2) (d) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop** | **CHAPTER 13 SERIAL INTERFACE IICA** |
| | Modification of **13.5.17 (3) (d) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop** | |
| | Modification of **Figure 14-1. Block Diagram of Multiplier and Divider/Multiply-Accumulator** | **CHAPTER 14 MULTIPLIER AND DIVIDER/MULTIPLY-ACCUMULATOR** |
| | Modification of **caution 1** to **14. 2 (2) Multiplication/division data register B (MDBL, MDBH)** | |
| | Modification of **caution 2** to **14. 2 (3) Multiplication/division data register C (MDCL, MDCH)** | |
| | Modification of description in **Figure 14-5. Format of Multiplication/Division Control Register (MDUC)** | |
| | Modification of description in **14.4.1 Multiplication (unsigned) operation,** and modification of value in **Figure 14-6. Timing Diagram of Multiplication (Unsigned) Operation (2 × 3 = 6)** | |
| | Modification of description in **14.4.2 Multiplication (signed) operation,** and modification of value in **Figure 14-7. Timing Diagram of Multiplication (Signed) Operation (−2 × 32767 = −65534)** | |
| | Modification of description in **14.4.3 Multiply-accumulation (unsigned) operation** | |
| | Modification of value in **Figure 14-8. Timing Diagram of Multiply-Accumulation (Unsigned) Operation** | |
| | Addition of description to **14.4.4 Multiply-accumulation (signed) operation** | |
| | Modification of value in **Figure 14-9. Timing Diagram of Multiply-Accumulation (signed) Operation** | |
| | Modification of description in **14.4.5 Division operation** | |
| | Modification of value in **Figure 14-10. Timing Diagram of Division Operation (Example: 35 ÷ 6 = 5, Remainder 5)** | |
| | Addition of description | **CHAPTER 16 INTERRUPT FUNCTION** |

(6/8)

| Edition | Description | Chapter |
|---|---|---|
| Rev.0.03 | Deletion of **caution 2** in **Figure 16-2. Format of Interrupt Request Flag Registers (IF0L, IF0H, IF1L, IF1H, IF2L, IF2H, IF3L) (128-pin)** | **CHAPTER 16 INTERRUPT FUNCTION** |
| | Modification of value and addition of **note** in **Table 16-4. Time from Generation of Maskable Interrupt Until Servicing** | |
| | Modification of **Figure 16-8. Interrupt Request Acknowledgment Timing (Minimum Time)** and **Figure 16-9. Interrupt Request Acknowledgment Timing (Maximum Time)** | |
| | Modification of **Table 16-5. Relationship Between Interrupt Requests Enabled for Multiple Interrupt Servicing During Interrupt Servicing** | |
| | Deletion of **caution** in **16.4.4 Interrupt request hold** | |
| | Change of all | **CHAPTER 17 KEY INTERRUPT FUNCTION** |
| | Addition of **note 1** and modification of **note 2** in **Figure 18-3. HALT Mode Release by Interrupt Request Generation** | **CHAPTER 18 STANDBY FUNCTION** |
| | Modification of description and **note** in **Figure 18-4. HALT Mode Release by Reset** | |
| | Modification of description in **Table 18-2. Operating Statuses in STOP Mode** | |
| | Modification of **note** in **Figure 18-5. STOP Mode Release by Interrupt Request Generation** | |
| | Modification of **note** in **Figure 18-6. STOP Mode Release by Reset** | |
| | Modification of description in **18.3.3 (1) SNOOZE mode setting and operating statuses** | |
| | Modification of description in **Table 18-3. Operating Statuses in SNOOZE Mode** | |
| | Modification of **Figures 19-2** to **19-4** | **CHAPTER 19 RESET FUNCTION** |
| | Modification of description in **Table 19-1. Operation Statuses During Reset Period** | |
| | Modification of **note 2** in **Table 19-2. Hardware Statuses After Reset Acknowledgment** | |
| | Modification of description and **notes** in **Figure 20-2. Timing of Generation of Internal Reset Signal by Power-on-reset Circuit and Voltage Detector** | **CHAPTER 20 POWER-ON-RESET CIRCUIT** |
| | Modification of **Figure 21-1. Block Diagram of Voltage Detector** | **CHAPTER 21 VOLTAGE DETECTOR** |
| | Modification of description in **Figure 21-2. Format of Voltage Detection Register (LVIM)** | |
| | Addition of figure to **Table 21-1. LVD Operation Mode and Detection Voltage Settings for User Option Byte (000C1H)** | |
| | Modification of **Figures 21-4** to **21-6** | |
| | Addition of description and **Figure 21-7, 21-8** to **21.4.3 When used as interrupt and reset mode** | |
| | Modification of **remark** in **22.1 Overview of Safety Functions** | **CHAPTER 22 SAFETY FUNCTIONS** |
| | Addition of description and **caution** to **22.3.1 Flash memory CRC operation function (high-speed CRC)** | |
| | Modification of **Figure 22-3. Flowchart of Flash Memory CRC Operation Function (High-speed CRC)** | |
| | Addition of description and **caution** to **22.3.2 CRC operation function (general-purpose CRC)** | |

(7/8)

| Edition | Description | Chapter |
|---|---|---|
| Rev.0.03 | Modification of **Figure 22-6. CRC Operation Function (General-Purpose CRC)** | **CHAPTER 22 SAFETY FUNCTIONS** |
| | Modification of **caution** in **Figure 22-7. Format of RAM Parity Error Control Register (RPECTL)** | |
| | Addition of **remark** to **Figure 22-11. Format of Invalid Memory Access Detection Control Register (IAWCTL)** | |
| | Addition of description to **22.3.8 A/D test function** | |
| | Addition of Figure (move from 2.2 Description to Pin Function (preceding editions)) | **CHAPTER 23 REGULATOR** |
| | Modification of description in **Figure 24-1. Format of User Option Byte (000C0H/010C0H)** | **CHAPTER 24 OPTION BYTE** |
| | Modification of **Figure 24-2. Format of User Option Byte (000C1H/010C1H)** | |
| | Modification of description in **25.1.1 Programming Environment** and modification of **Notes** in **25.1.2 Communication Mode** | **CHAPTER 25 FLASH MEMORY** |
| | Modification of **note 1** and addition of **note 2** in **Table 25-2. Pin Connection**<br>Addition of description to **25.2 Writing to Flash Memory by Using External Device (that Incorporates UART)** | |
| | Modification of note in **25.2.2 Communication Mode** | |
| | Addition of **remark** to **25.3 Connection of Pins on Board** | |
| | Modification of description and addition of **remark** to **25.4.1 Data flash overview** | |
| | Modification of **Figure 25-8. Setting of Flash Memory Programming Mode** | |
| | Modification of **Table 25-5. Programming Modes and Voltages at Which Data Can Be Written, Erased, or Verified** | |
| | Modification of description in **Table 25-10. Example of Signature Data** | |
| | Addition of description and **caution** to **25.6 Security Settings** | |
| | Addition of **25.7 Processing Time of Each Command When Using PG-FP5 (Reference Values)** | |
| | Addition of **cautions 3, 4** to **25.8 Flash Memory Programming by Self-Programming** | |
| | modification of value in **Table 25-14. Programming Modes and Voltages at Which Data Can Be Written, Erased, or Verified** | |
| | Modification of Flag status | **CHAPTER 28 INSTRUCTION SET** |
| | Addition of **cautions 2, 3** to **CHAPTER 29 ELECTRICAL SPECIFICATIONS** (deletion of Pins Mounted According to Product) | **CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C) (TARGET)** |
| | Addition of description, **note 3**, and **remark 2** to **29.1 Absolute Maximum Ratings** | |
| | Addition of **note 2** to **29.2.2 On-chip oscillator characteristics** | |
| | Modification of **note** in **29.3.2 Supply current characteristics** | |
| | Addition of figure to **29.4 AC Characteristics** | |
| | Deletion of description and value to **29.5.1 Serial array unit** | |
| | Addition of value to **29.5.2 Serial interface IICA** | |
| | Modification of **29.6.1 A/D converter characteristics** | |
| | Modification of **29.6.2 Temperature sensor characteristics** and **29.6.3 POR circuit characteristics** | |

(8/8)

| Edition | Description | Chapter |
|---|---|---|
| Rev.0.03 | Modification of description and addition of **caution** in **29.6.4 LVD circuit characteristics** | **CHAPTER 29 ELECTRICAL SPECIFICATIONS ($T_A$ = –40 to +85°C) (TARGET)** |
| | Modification and addition of **note 3** in **29.8 Flash Memory Programming Characteristics** | |
| | Modification of description in **29.9 Timing Specs for Switching Flash Memory Programming Modes** | |
| | Addition of **CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS TA = –40 to +105°C) (TARGET)** | **CHAPTER 30 ELECTRICAL SPECIFICATIONS (G: INDUSTRIAL APPLICATIONS $T_A$ = –40 to +105°C) (TARGET)** |
| | Addition of the products of industrial application | **CHAPTER 31 PACKAGE DRAWINGS** |
| Rev.0.02 | Change of **Table 2-3. Connection of Unused Pins (64-pin products)** | **CHAPTER 2 PIN FUNCTIONS** |
| | Change of **Figure 11-4. Timing Chart When A/D Voltage Comparator Is Used** | **CHAPTER 11 A/D CONVERTER** |
| | Change of **Table 11-3. A/D Conversion Time Selection** | |
| | Deletion of **caution 3** in **Figure 11-6. Format of A/D Converter Mode Register 1 (ADM1)** | |

RL78/G1A User’s Manual: Hardware

Publication Date: Rev.2.00 Jul 04, 2013

Published by: Renesas Electronics Corporation

**SALES OFFICES** Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.


**Renesas Electronics America Inc.**
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

**Renesas Electronics Canada Limited**
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

**Renesas Electronics Europe Limited**
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700, Fax: +44-1628-651-804

**Renesas Electronics Europe GmbH**
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

**Renesas Electronics (China) Co., Ltd.**
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

**Renesas Electronics (Shanghai) Co., Ltd.**
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

**Renesas Electronics Hong Kong Limited**
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

**Renesas Electronics Taiwan Co., Ltd.**
13F, No. 363, Fu Shing North Road, Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

**Renesas Electronics Singapore Pte. Ltd.**
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

**Renesas Electronics Malaysia Sdn.Bhd.**
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

**Renesas Electronics Korea Co., Ltd.**
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141



Colophon 1.3

RL78/G1A

Renesas Electronics Corporation

R01UH0305EJ0200